NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

ENHANCING AUTONOMY OF AERIAL SYSTEMS VIA INTEGRATION OF VISUAL SENSORS INTO THEIR AVIONICS SUITE

by

Kenny Teo

September 2016

Thesis Advisor:	Oleg Yakimenko
Second Reader:	Fotis Papoulias

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2016	**3. REPORT TYPE AND DATES COVERED** Master's thesis	
4. TITLE AND SUBTITLE ENHANCING AUTONOMY OF AERIAL SYSTEMS VIA INTEGRATION OF VISUAL SENSORS INTO THEIR AVIONICS SUITE			**5. FUNDING NUMBERS**
6. AUTHOR(S) Kenny Teo			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200words)

Autonomous aerial systems have started to gain much traction in the military intelligence, surveillance and reconnaissance domain. The remotely piloted systems, such as the Predator, are already successful unmanned systems; the next step forward is to use autonomous systems to overcome high manning requirements. These systems are scalable and serve as excellent force multipliers, but there are other technological issues to overcome to qualify an autonomous aerial system, such as navigation and collision avoidance.

This thesis explores autonomous system capabilities using quadrotors in the context of the Singapore Armed Forces. It first applies a systems engineering approach to analyze stakeholders' needs, then translates the needs to functional requirements, and concludes with the development of a possible system architecture for an autonomous quadrotor system. The author then conducted indoor flight experiments to validate the capabilities of waypoint navigations and collision avoidance. The results were highly encouraging and qualified the aerial platform for subsequent visual sensor integration.

14. SUBJECT TERMS autonomous system, quadrotors, direct method, inverse dynamics, virtual domain			**15. NUMBER OF PAGES** 111
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

ENHANCING AUTONOMY OF AERIAL SYSTEMS VIA INTEGRATION OF VISUAL SENSORS INTO THEIR AVIONICS SUITE

Kenny Teo
Captain, Singapore Army
B.Eng., National University of Singapore, 2012

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2016

Approved by: Oleg Yakimenko
Thesis Advisor

Fotis Papoulias
Second Reader

Ronald Giachetti
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Autonomous aerial systems have started to gain much traction in the military intelligence, surveillance and reconnaissance domain. The remotely piloted systems, such as the Predator, are already successful unmanned systems; the next step forward is to use autonomous systems to overcome high manning requirements. These systems are scalable and serve as excellent force multipliers, but there are other technological issues to overcome to qualify an autonomous aerial system, such as navigation and collision avoidance.

This thesis explores autonomous system capabilities using quadrotors in the context of the Singapore Armed Forces. It first applies a systems engineering approach to analyze stakeholders' needs, then translates the needs to functional requirements, and concludes with the development of a possible system architecture for an autonomous quadrotor system. The author then conducted indoor flight experiments to validate the capabilities of waypoint navigations and collision avoidance. The results were highly encouraging and qualified the aerial platform for subsequent visual sensor integration.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

6DOF	six degrees of freedom
AI	artificial intelligence
AO	area of operation
ASEIL	Autonomous Systems Engineering and Integration Laboratory
BIT	built-in test
DAQ	Data acquisition card
DOD	Department of Defense
DODAF	DOD Architectural Framework
DRM	Design reference mission
EO	electro-optical
FSTD	Future Systems and Technology Directorate
GCS	ground control station
GPS	Global-Positioning System
IDVD	inverse dynamics in the virtual domain
ILP	integer linear program
INS	inertial-navigation system
IP	Internet protocol
IR	infrared
ISR	intelligence, surveillance and reconnaissance
JP	joint publication
JPEG	joint photographic experts group
LIDAR	light detection and ranging
LQR	linear quadratic regulator
MLRS	multiple-launch rocket system
NP	non-deterministic polynomial time
OV	operational view
POE	Projected Operational Environment
PWM	pulse-width modulation
RGB	red green blue
ROE	rules of engagement

SAF	Singapore Armed Forces
STORM	Strike Observers Mission
SV	system view
TFR	total fertility rate
TRL	technology readiness level
TSP	traveling salesman problem
UAV	unmanned aerial vehicle
USB	universal serial bus
VTOL	vertical takeoff and landing

EXECUTIVE SUMMARY

The Singapore Armed Forces faces an imminent manning crunch, as the number of conscripts enlisted per year is likely to drop to 15,000 from 21,000 (Saad 2012). This is a result of the downward trend in the nation's total fertility rate. As the standing defense force becomes leaner, it is imperative that the SAF exploits technological advances to develop highly effective systems that require fewer operators in achieving the ultimate mission of securing swift and decisive victories should diplomacy fail. The concept of autonomous aerial systems appears a feasible and logical solution in maintaining precision-strike capabilities for the SAF, which can be labor intensive, especially in target acquisition. An autonomous aerial system with onboard visual sensors may offer target acquisition with reduced manning. Moreover, the system is potentially scalable and so many machines to one operator ratio is highly achievable.

The systems engineering approach was adopted in this research to determine a potential system architecture for such an autonomous aerial system. The author first defined the boundary of the design space the context of the problem space. Thereafter, the thesis investigated the limitations and constraints that will impose on the system. Following which, the author performed the needs analysis on the relevant stakeholders of the system, which is vital in deriving the functional requirements. In addition, the thesis also elaborated on the design reference mission and operational concept to provide the operational context of the system. These steps contributed to the formulation of a proposed system architecture for an autonomous aerial system with on-board visual sensors, capable of cooperative target acquisition.

The proposed autonomous aerial system recommends optimized search paths to save resources such as exposure time and fuel. Not everywhere in the area of search is logical to detect a target. For example, the system ignores bodies of water during detection of a land target in flight-path optimization. Using a pre-processed map provided by the operator, the system reads the feasible search area and performs an optimized search using binary integer linear programming (ILP). The output is a series of waypoints arranged sequentially for the aerial system to follow. From a simulation performed in an

area of operation with a non-feasible search area of approximately 30%, the use of ILP achieved a savings of approximately the same ratio in distance traveled.

While waypoints may guide an autonomous aerial system navigationally, there are bound to be obstacles in the path of the system. Hence, the proposed autonomous aerial system is capable of collision detection and obstacle avoidance. The on-board visual sensors detect any imminent obstacles. With the known coordinates of the obstacles, the system generates a near-optimal trajectory real-time to overcome the obstacles. Cowling et al., (2010) has proposed the use of a form of direct method of calculus of variables in this trajectory optimization process. Crucial to the application of the direct method was exploitation of the inverse dynamics of the aerial system so that the optimization occurs in the output space rather than the control space. In addition, the method optimizes the variables in the virtual domain instead of the conventional time domain, decoupling the time and space parameterizations, whose combination tends to create complexity in resolving the variables. Collectively, the method used is inverse dynamics in the virtual domain (IDVD).

The Autonomous Systems Engineering and Integration Laboratory (ASEIL) in Naval Postgraduate School was where the author performed experiments to verify the optimized trajectories generated. The author conducted three verification tests using Qball-X4 quadrotors with modified controllers. With the simulated waypoints generated from the optimized flight path derived from the ILP, the Qball successfully performed waypoint navigation in the first verification test. In the second test, the Qball executed an overhead maneuver according to the near-optimal trajectory generated by the IDVD method, avoiding collision with a vertical obstacle. The third test verified the controller’s ability to mix guidance commands and execute flat and overhead trajectories interchangeably.

While this research was not able to integrate the visual sensor physically, owing problems with vendor software, the architecture and control algorithm developed stand ready for such integration. It is recommended that future work focus on the physical integration of the visual sensor to validate the autonomy of proposed aerial systems.

References

Cowling, Ian D., Oleg A. Yakimenko, James F. Whidborne, and Alastair, K. Cooke. 2010. “Direct Method Based Control System for an Autonomous Quadrotor.” *Journal of Intelligent and Robotic Systems* 60 (2): 285–316. doi:10.1007/s10846-010-9416-9.

Saad, Imelda. 2011. “SAF Uses Technology to Counter Lower Birth Rate & Ageing Population.” *Channel NewsAsia*, November 12. http://www.channelnewsasia.com/news/singapore/saf-uses-technology-to-co/514780.html.

ACKNOWLEDGMENTS

I extend my heartfelt gratitude to Professor Oleg Yakimenko for his invaluable patience and guidance. My sincere appreciation also goes to Professor Fotis Papoulias for his sound advice.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. THE UAV AS A FORCE MULTIPLIER

Like most developed nations, Singapore is experiencing a declining total fertility rate (TFR), which began in the early 1990s. According to the Singapore Department of Statistics (2015), the TFR in 2014 was 1.25, significantly below the replacement level of approximately 2.1 children per woman. Sustained by conscription, the Singapore Armed Forces (SAF) relies on the civilian population to form the standing defense force. In 2011, the incumbent Minister of Defense, Dr. Ng Eng Hen reported 21,000 enlisted conscripts (Saad 2012). He added that this number could decrease to 15,000 and possibly lower in the future. As the standing defense force grows leaner, it is imperative that the SAF exploits technological advances to assure highly effective systems that require fewer operators while still achieving the ultimate mission of securing swift and decisive victories should diplomacy fail. The concept of autonomous aerial systems appears to be a feasible and logical solution in maintaining SAF precision-strike capabilities, which are typically labor intensive, particularly in target acquisition. An autonomous aerial system with onboard visual sensors may prove effective in performing target acquisition with reduced manning. Moreover, the system is potentially scalable and so many machines to one operator ratio is highly achievable. The unmanned aerial vehicle (UAV) is an example of a force multiplier used by the SAF.

UAVs first played a sustained operational role in the Vietnam War (Cook 2007). Largely functioning within the intelligence, surveillance and reconnaissance (ISR) domain, mission commanders used UAVs in place of manned reconnaissance aircrafts because of their ability to perform "missions deep within the enemy territories at a fraction of the cost" (Cook 2007, 3). In a high-threat environment, the employment of UAVs reduces the risk of human casualties. According to the Office of the Secretary of Defense, as of September 2004, UAVs have clocked over 100,000 of flight hours in support of Operation Enduring Freedom and Operation Iraqi Freedom (2005). Furthermore, the Department of Defense (DOD) increased its spending on UAVs by more than tenfold between 2000 and 2012 (Boyle 2012), and UAVs now account for approximately a third of DOD military aircraft (Shachtman and Akerman 2012). While strike capability is possible through weaponized UAVs, as demonstrated by the Predators, the UAV's most common role is within the realms of target acquisition.

B. TARGET ACQUISITION

The Joint Chiefs of Staff, in Joint Publication (JP) 3-60, defines "target acquisition" as "the detection, identification, and location of a target in sufficient detail to permit the effective employment of weapons" (2007). Target acquisition can be a daunting task. Knowing where to look for the adversary is a huge intelligence issue. To detect, identify and pinpoint the location of the target in an area of operation that extends for miles requires a great many assets. Nevertheless, such an effort is vital in the responsible strike operations expected of a world-class military today, and the SAF has invested tremendous resources in remaining ready and relevant in the future battlefield.

Strike-observers mission (STORM) teams are one of the critical assets the SAF uses to achieve third-generation fighting capabilities. As depicted in Figure 1, a STORM Team is a forward sensor unit that detects and destroys targets from a mobile platform. The sensors carried by the team include UAVs for long-range detection and handheld binoculars for short-range location of targets. STORM teams enhance SAF's effectiveness by neutralizing key targets such as command posts, artillery, and multiple-launch rocket systems (MLRS). The force configuration of a typical STORM team includes two sensor specialists (Defence Media Centre 2008) to operate the UAV and control onboard sensors. It is not currently possible to trim these personnel. Thus, the number of pilots available restricts the overall target-acquisition capability. Staffing needs, however, have potential for reduction if autonomous machines can perform functions or operations that have historically required a human.

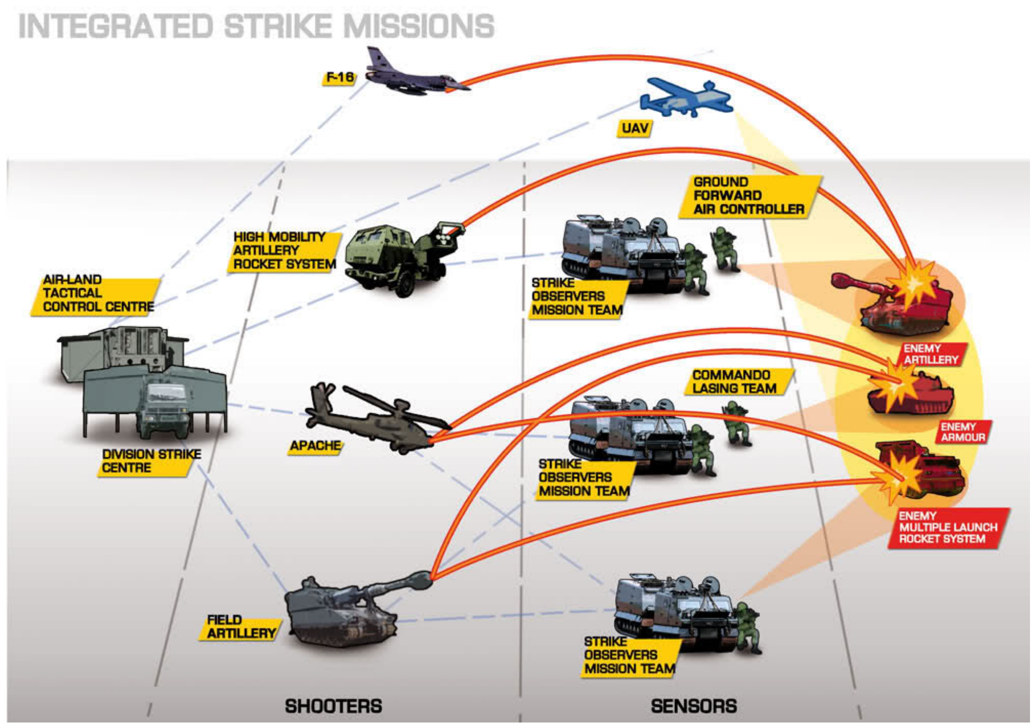
INTEGRATED STRIKE MISSIONS
F-16
UAV
GROUND FORWARD AIR CONTROLLER
HIGH MOBILITY ARTILLERY ROCKET SYSTEM
AIR-LAND TACTICAL CONTROL CENTRE
STRIKE OBSERVERS MISSION TEAM
ENEMY ARTILLERY
COMMANDO LASING TEAM
ENEMY ARMOUR
APACHE
DIVISION STRIKE CENTRE
STRIKE OBSERVERS MISSION TEAM
ENEMY MULTIPLE LAUNCH ROCKET SYSTEM
FIELD ARTILLERY
STRIKE OBSERVERS MISSION TEAM
SHOOTERS
SENSORS

Figure 1. Role of STORM in SAF Integrated Strike Missions. Adapted from Defence Media Centre (2011).

C. AUTONOMOUS OPERATIONS

The term UAV has often been misinterpreted. Commonly known UAVs such as the Global Hawk and Predator are not exactly unmanned systems. There is still a pilot remotely controlling the vehicle, and thus the man-to-machine ratio remains one. Lin et al. propose that an autonomous machine has “the capacity to operate in the real-world environment without any form of external control, once the machine is activated and at least in some areas of operation, for extended periods of time” (2008, 4). Such machines could improve the man-to-machine ratio either by decreasing the number of operators required or increasing the number of machines a single operator can command. In either scenarios, the humans in the system can assume other roles, enhancing system capabilities overall. To a force that anticipates decreases in available staffing, the notion of autonomous operation via machine is compelling.

While the SAF does not currently employ autonomous militarized equipment, the possibility of operationalizing an autonomous system to fulfill command intentions may be within reach. Colonel Chua, SAF (Retired), who heads the Concept and Experimentation Office of the Future Systems and Technology Directorate (FSTD), anticipates that future warfare will see “many close interactions between soldiers and autonomous robots” (Teo 2016). This cooperation between man and autonomous machines will greatly improve the situational awareness and lethality of the SAF while minimizing danger to troops.

D. QUADROTORS VS. FIXED-WING UAVS

Fixed-wing aircrafts offer the user with high speed and long endurance, but their inability to fly at low speed or to hover above a locality disqualifies them from certain target-acquisition missions. Thus, the notion of using rotary aircraft has gained popularity. Rotary aircraft allows hovering, which provides persistent target surveillance without a large operational footprint, and vertical takeoff and landing (VTOL), which shrinks the area required for launch and recovery and obviates the need for a launcher or runway.

A traditional rotary aircraft has a main rotor that provides lift and pitch and a tail rotor to counter reactionary torque. Because the cyclic-pitch mechanism of the main rotor is mechanically complex and potentially expensive to maintain, quadrotors appears as an attractive alternative. With a pair of counter-rotating propellers connected directly to individual motors, the quadrotor has better mobility and higher payload capacity at lower maintenance cost. The disadvantages of quadrotors include sensitivity to disturbances and dynamic instability—hence their need for precise control implementation. Additionally, because the four rotors imply more motors and greater weight, quadrotors consume more power for operation.

The use of fixed-wing UAVs to conduct ISR missions is commonplace, but that is not to say that quadrotors in the surveillance realm are unheard of. For example, Aeryon Labs, a Canadian company, claims that an Aeryon Scout quadrotor’s visual surveillance of a narco-trafficking compound deep in the Central American jungle was crucial in a successful drug bust (Guizzo 2011).

E. PROBLEM FORMULATION AND THESIS STRUCTURE

This thesis examines how quadrotors can achieve autonomous target-acquisition capability and addresses several problems in the fulfillment of this role.

First, at the mission-planning phase, there is a need to conduct flight-path planning. The quadrotor is nothing more than a platform that executes the instructions of the mission commander. It has no artificial intelligence or ability to plan its flight path. A human commander must perform this planning action. The necessary inputs for the autonomous target-acquisition system are the waypoints along the intended path and the tasks to execute. While a human can systematically plot waypoints for a quadrotor to follow, the vehicle does not necessarily cover the entire area of operation efficiently. There is a need for effective task performance to save resources such as time of asset exposure.

Next, during the execution phase, there bound to be obstacles along the path of the autonomous system. These obstacles are likely man-made structures, such as buildings, radio towers or overhead bridges. For the survival of the aircraft, it is imperative that the quadrotor avoids, at the very least, the known and foreseeable obstacles. Therefore, it calls for some form of trajectory correction to avoid collision and to overcome the obstacles.

Finally, at the heart of the autonomous system is target acquisition using on-board electro-optical (EO) sensors. While many advanced sensors are available, EO imagery is the basic means of target acquisition, because the output is congruent with the human vision, and allows for quick user interpretation and intervention when necessary. This research therefore also investigates the incorporation of EO output into autonomous target-acquisition capability.

The organization of this thesis as follows: Chapter II explores the needs and functions of an autonomous quadrotor using a systems engineering approach. Chapter III presents a modeling and simulation environment for the validation of these capabilities. Chapter IV discusses the actions required to plan and optimize a search path. Chapter V addresses trajectory generation, using a form of direct method to overcome vertical obstacles. Chapter VI explores the control scheme required to mix position commands in a quadrotor controller. Chapter VII presents the results of simulations conducted. This thesis ends with conclusions and recommendations to study how to further achieve autonomous capabilities using quadrotors.

While it is outside the scope of this research to verify and validate quadrotor target acquisition and collision detection through real-time image processing, the latter is broached in Appendix A to suggest the feasibility on visual-sensor integration and set the stage for future work.

II. SYSTEMS ENGINEERING CONSIDERATIONS

While a big body of literature explored the application space of the quadrotors already, not many research efforts have defined the system architecture of the quadrotors, especially as pertains to the use of quadrotors in autonomous target acquisition. This chapter applies a systems-engineering approach to this question.

A. BENEFITS OF A SYSTEMS ENGINEERING APPROACH

> Systems engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem. (Haskins et al., 2006)

Systems engineering is a discipline vital to the development of complex systems in many industries. Systems thinking is an integral part of this approach. According to Honour, an emphasis on systems thinking within a program development yields easier and faster testing and integration of subsystems (2004), and the outcome of the application is typically a high-quality system product achieved with a shorter program schedule and lower cost. Honour argues that at the core of the systems-engineering approach is the goal of reducing risk early on; systems engineers identify the challenges of testing and integration up front and attempt to resolve them in the design phase. This reduces cost and time in the production and testing phases.

B. DEFINITION OF A DESIGN SPACE

As the SAF addresses the anticipated dearth of conscripts, the nation’s advanced technological capabilities and expertise provide an excellent basis for exploring autonomy in the realm of ISR, specifically in target acquisition.

The problem this research seeks to resolve is how to use quadrotors in achieving autonomous target-acquisition. The questions stemming from this key problem are a) which system architecture is best suitable and b) what level of autonomy can the system support?

1. Boundaries

Allowing autonomous systems to operate alone in the battlefield represents a huge paradigm shift, challenging expectations and physical, economic, technological, political, and psychological boundaries.

The physical boundary encompasses natural obstacles, ambient environmental factors such as temperature and wind speed, and the operating environment of the autonomous system. The projected operating environment is discussed in a later section.

The economic boundary is the limit to the monetary resources allocated for the system. As the development costs of an autonomous system are high, the stakeholders responsible for developing the system must be mindful in their expenditures.

The technology readiness level (TRL) of the various subsystems forms the technological boundary. Challenges to system integration occur when mismatches in the TRLs of subsystems are present. TRL also affects the developmental funding awarded to the autonomous system, given that conducting research and developing immature technology is expensive.

The international rules that bind every legitimate country largely define the political boundary. In a conflict, the international laws of war and rules of engagement (ROE) apply. These rules protect non-combatants and safeguard fundamental human rights.

The intangible perception of autonomous systems constitutes the psychological boundary. As mentioned earlier, it is a paradigm shift when thinking machines work alongside soldiers in armed conflict. The public may be wary of their potential, and the soldiers may doubt their capabilities. Fear of an autonomous system's losing control and triggering catastrophic outcomes will not be easy to eradicate.

2. Limitations

The critical limiting factor for an autonomous system is the communication network between the system itself and other command-transmitting stations. Ground control stations (GCSs) have difficulty transmitting commands to an autonomous system over long distances, especially where there is no line of sight. An autonomous system may have difficulty in receiving Global Positioning System (GPS) satellite signals in dense vegetation or an urban environment. Interference or jamming from other electronic sources may also hinder communication paths.

The bandwidth capacity of participating communication paths is another limiting factor. Given the need to transit videos and imageries, high data throughput and fidelity are essential. Existing communication infrastructure may not be able to support an autonomous system.

3. Constraints

Lack of airspace is a significant constraint in autonomous systems, which require freedom of action in the air to perform target acquisition. Enemy aerial-denial weapons in the operating environment may potentially diminish the overall capability of a system.

Safety in operating the autonomous system is a major consideration as well; soldiers will need training to operate this unfamiliar system without harm. Moreover, because conscripts constitute the bulk of SAF personnel, the unsafe operation of an unproven system would compromise public confidence and support for the military.

Finally, the boundaries discussed above act as restrictive constraints for the autonomous system as well.

C. THE NEEDS ANALYSIS

Stakeholders may affect the system directly or indirectly. Their differing needs and perspectives are the drivers of system requirements.

1. The Main Stakeholder: SAF

The predominant stakeholder, the SAF, is concerned with manpower requirements and mission success. The SAF's ability to deter aggression is crucial to its posture as a military force—hence the imperative of leveraging technology to compensate for shortfalls in manpower. Despite a lean force, the SAF must remain capable of maintaining the same or higher lethality to secure swift and decisive victories should deterrence fail. In addition, the SAF is also the key architect in integrating systems together to fight as a cohesive force. It is therefore important to ensure interoperability between any autonomous system and other existing operational systems.

2. Operators

The operators on the ground are the first-hand stakeholders. They are vitally concerned with the human–machine interfaces that affect workflow. These users require an easy to operate, maintain, and troubleshoot system.

3. Strike Units

The artillery battalions or fighter squadrons who remove threats posed by the adversary in strike missions represent the strike units. These units require real-time and accurate information to prosecute the target. They are concerned with how target information arrives as this affects their work processes.

4. The Adversary

It may not be intuitive to count the adversary as a stakeholder in the target-acquisition system. Nevertheless, his ability to evade detection directly drives system requirements. While tactics evolve, the form and capabilities of enemy assets remain. Prior intelligence on these assets provides understanding as to how best to detect them.

5. The Public

The public is now able to access, and even capture, footage of wartime operations, which may move through social media at an exponential speed. In times of conflict, the need to ensure that soldiers and systems adhere to the defined ROE is a top priority. In peacetime, the public is largely concerned with total military expenditures, as public monies fund the defense budget. Responsible spending on research and equipment is imperative in developing the system.

D. OPERATIONAL CONCEPT

The DOD architectural framework (DODAF) model operation view 1, or OV-1, shown in Figure 2 best illustrates the operational concept for an autonomous target acquisition system. In this framework, the operator inputs the necessary command parameters, which typically consist of search area and type of target, into the GCS. The autonomous system computes the optimal flight path for the given area of search. Upon receipt of the command to commerce operation, the quadrotors launch and search autonomously, leveraging GPS to navigate along the trajectories given. Using a wireless camera and onboard computer, the quadrotors perform image processing in real time. Once the quadrotor finds the target, it acquires its location and transmits this data to the operator. The quadrotor continues to loiter in the target vicinity to provide continuous observation and can designate the target to mark it for follow-on strike operations.

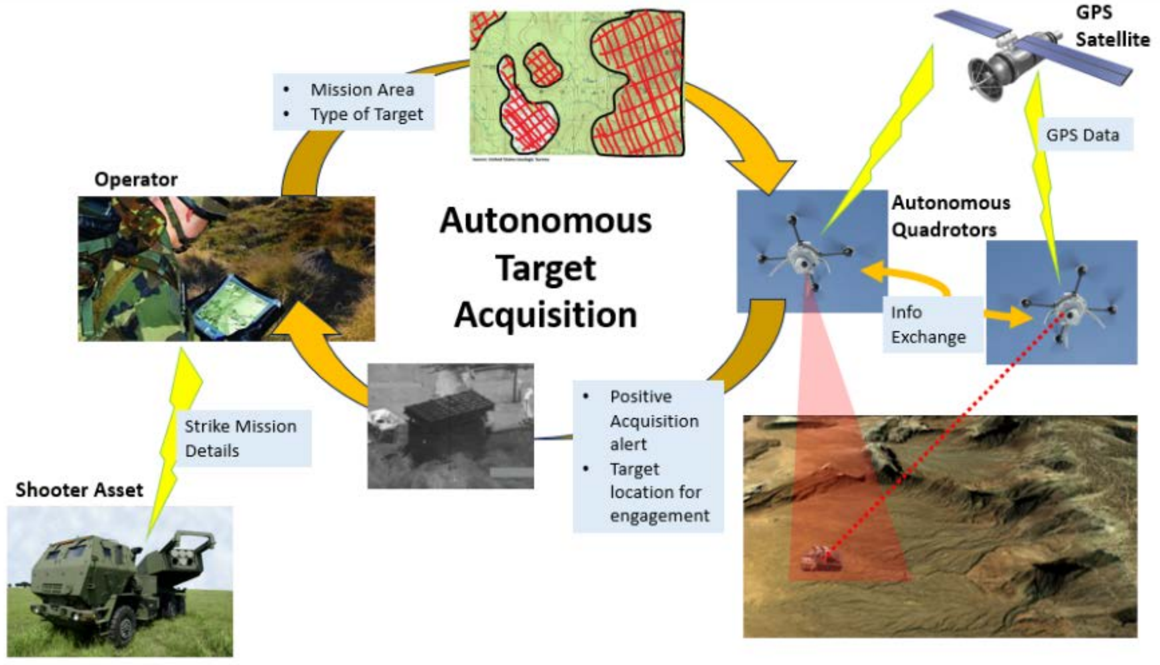

Figure 2. OV-1 for Autonomous Target Acquisition

E. DESIGN REFERENCE MISSION

Skolnick and Wilkins define a design reference mission (DRM) as the "projected threat and operating environment baseline" by which the desired system meets potential challenges and uncertainties through a series of systems-engineering activities (2000, 208). In this research, the DRM serves as a basis from which further requirements, system designs, and performance measures are developed.

1. Projected Operational Environment

The projected operation environment (POE) considers aspects of the natural environment and potential threats to the system. As this system is intended for SAF use, the POE is described in the context of Singapore.

a. Natural Environment

- The lighting conditions of the POE range from full sunlight to minimal outdoor moonlight at night.
- The wind condition of the POE ranges from 0–27 knots. As Singapore may experience the occasional "Sumatra Squalls," the upper limit of the wind condition is consistent with the upper bound of the Beaufort scale's "strong breeze."
- The temperature of the POE ranges from 68°F (20°C) to 97°F (36°C), which are the extreme temperatures recorded in Singapore.
- The humidity of the POE ranges from 60% to 100%.
- A quadrotor may operate near water bodies or in rain. Therefore, while it does not carry out submerged operations, the system must remain operable in depths of up to 0.5m and in light rainfall where precipitation is less than 2.5mm (0.098in) per hour. The system must also be splash resistant.

b. Threat Details

Two broad categories generally faced by the quadrotor in the POE are natural or man-made. The threats cited as follows are mainly generic and not exhaustive.

(1) Natural

- Trees

- Dust and dirt
- Water

(2) Human

- Man-made structures
- Personnel
- 5.56mm/7.62mm rounds from enemy short-range weapons
- Destruction of GCS by enemy long-range weapons

2. Mission Definition

The mission of the quadrotor is to support the STORM team in autonomous target acquisition to facilitate follow-on strikes. Mission success is defined by achievement of an optimized search path, high fidelity in the commanded-versus-actual flight path, accurate target acquisition, and long-term cooperative tracking of the target.

In the course of a mission, the autonomous system must carry out several critical operational activities, which drive the functional baseline and shape the system architecture. These include

- Integrated network command processing
- Autonomous launch and recovery
- Autonomous guidance and control
- Autonomous target acquisition

F. FUNCTIONAL ANALYSIS

According to Blanchard and Fabrycky, “functional analysis is an iterative process that examines system requirements to yield detailed design criteria” (2011, 86). The process identifies the resources needed to support and operate the system.

With a review of the operational context and stakeholders’ needs, one can define the vital functions of an autonomous target-acquisition system. Figure 3 depicts the functional decomposition of the system and Table 1 captures functional descriptions.

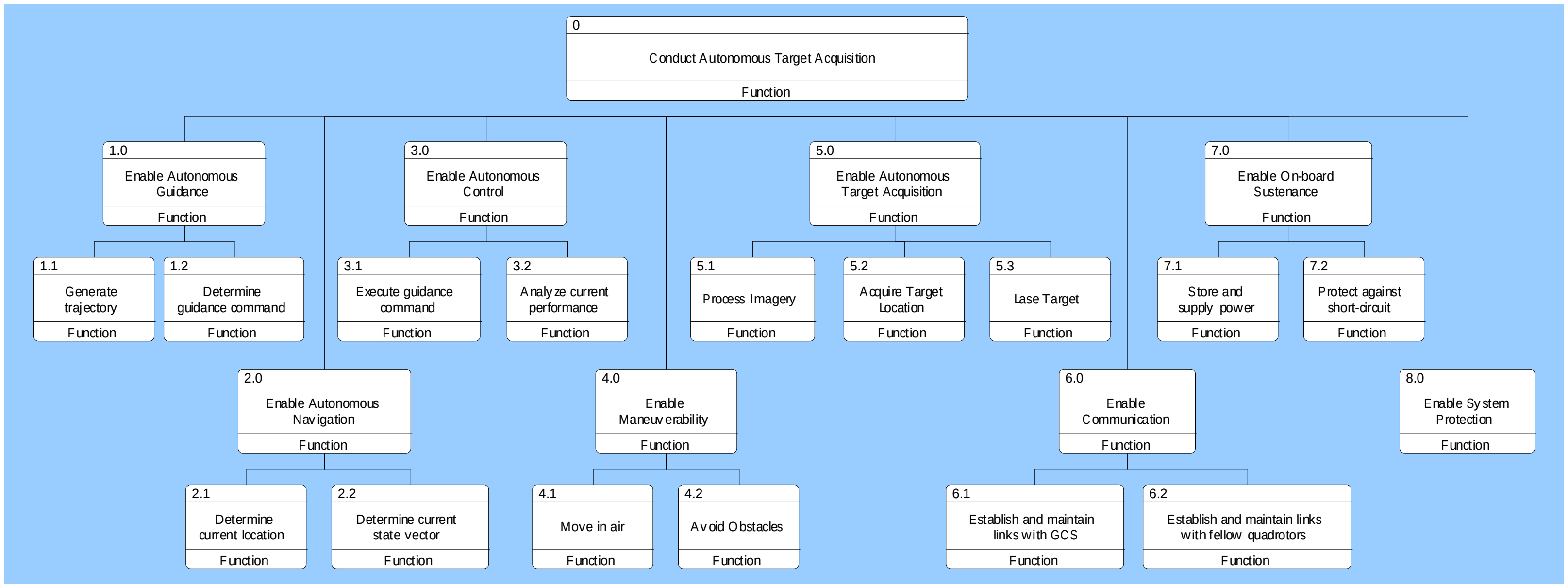

Figure 3. Functional Decomposition for the Conduct of Autonomous Target Acquisition

Table 1. Description of Functions for the Conduct of Autonomous Target Acquisition

S/N	Function	Description
0	Conduct Autonomous Target Acquisition	Top level function of the system
1.0	Enable Autonomous Guidance	The system level function of guiding the system autonomously.
1.1	Generate trajectory	The capability to generate an optimized trajectory given a pre-processed area of operation
1.2	Determine guidance command	The capability to determine the guidance commands, e.g., speed of rotors, to adhere to the desired path of travel
2.0	Enable Autonomous Navigation	The system level function of navigating the system autonomously.
2.1	Determine current location	The capability to determine the current location of the system
2.2	Determine current state vector	The capability to determine the current velocity or state vector of the system
3.0	Enable Autonomous Control	The system level function of controlling the system autonomously.
3.1	Execute guidance command	The capability to execute the guidance commands to achieve the necessary pitch, roll and yaw of the aircraft
3.2	Analyze current performance	The capability to analyze the current control performance of the system as part of feedback
4.0	Enable Maneuverability	The system level function of being able to maneuver in the airspace
4.1	Move in air	The capability to fly along the commanded trajectory
4.2	Avoid Obstacles	The capability to detect and avoid any obstacle in the flight path
5.0	Enable Autonomous Target Acquisition	The system level function of acquiring the desired target autonomously.
5.1	Process Imagery	The capability to process the captured imagery to check for desired target.
5.2	Acquire Target Location	The capability to acquire the coordinates of the target's location
5.3	Lase Target	The capability to lase the desired target for follow-on strike operation
6.0	Enable Communication	The system level function of communicating with external assets
6.1	Establish and maintain links with GCS	The capability to communicate with the GCS
6.2	Establish and maintain links with fellow quadrotors	The capability to communicate with the fellow quadrotors
7.0	Enable On-board Sustenance	The system level function of sustaining the power requirement of the system on-board.
7.1	Store and supply power	The capability to store and supply power to the entire system
7.2	Protect against short-circuit	The capability to protect the system from short-circuit
8.0	Enable System Protection	The system level function of protecting the system to enhance the survivability of the system

G. PROPOSED SYSTEM ARCHITECTURE

With the functional requirement in mind, this research proposes a system architecture for an autonomous target-acquisition quadrotor using the SV-1 model, as illustrated in Figure 4. The proposed architecture has seven subsystems, namely,

1. Processor and controller subsystem (depicted in yellow)
2. Flight actuators subsystem (green)
3. Target-acquisition subsystem (pink)
4. Communication subsystem (blue)
5. Positioning subsystem (purple)
6. Collision-avoidance subsystem (brown)
7. Power-supply subsystem (orange)

In the same model, the author describes the system interfaces by providing the required communications network for the link between subsystem components and the information exchanged through the stated link. The following subsections further describe these subsystems.

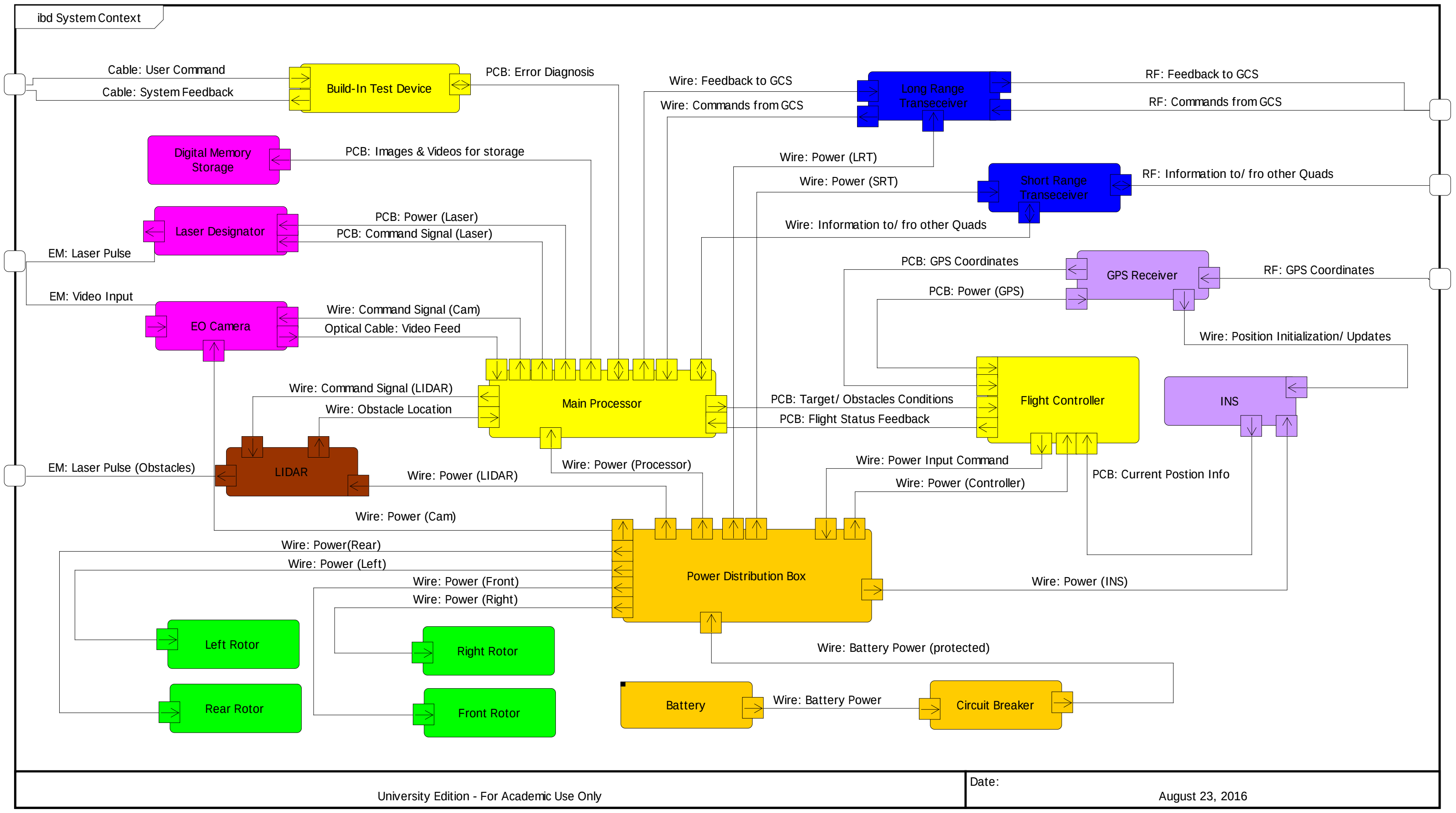

Figure 4. Proposed System Architecture for Autonomous Target Acquisition System Using Quadrotor

1. The Processor-and-Controller Subsystem

The proposed processor-and-controller subsystem contains three components: the main processor, the flight controller, and the built-in test (BIT) device. This subsystem acts as the brain of the quadrotor. While the main processor and the flight controller differ in nomenclature, they are essentially similar onboard computers separated to increase processing speed, as the quadrotor must execute computationally intensive algorithms rapidly in flight. The main processor is responsible for processing the imagery received from the EO camera, while the flight controller continuously optimizes the flight trajectory and issues control commands. The main processor also receives and processes commands from the GCS or other quadrotors in collaborative mode and commands the Light Detection and Ranging (LIDAR) sensor to pick up obstacles in the path. The main processor communicates information on any found target or obstacles to the flight controller, so that the latter can modify the existing flight trajectory by moving to a new vantage point to maintain persistent stare on the target or maneuvering away from the obstacle. The flight controller goes on to determine target location. The main processor receives this information from the flight controller and transmits it to the GCS. A deeper exploration of the architecture of a feasible flight controller is presented in Chapter V.

The BIT device is a component that runs built-in tests when the system boots up or on command. To save weight and space, there is no display interface for user interaction. The device receives or transmits all commands and feedback via wireless connection to the GCS or direct umbilical link to an external display and control.

2. The Flight-Actuators Subsystem

The flight-actuator subsystem comprises four rotors, with “rotor” understood as a set of motor, propeller and speed controllers. The rotors receive power inputs from the power-distribution box, and the thrust each rotor produces is a function of the power-input received. Varying the thrust of the rotors collectively produces flight motion—that is, elevation, roll, pitch, and yaw. Chapter VI discusses flight-control specifications.

3. The Target-Acquisition Subsystem

The target-acquisition subsystem consists of an EO camera, laser designator, and digital memory storage. The camera serves as the eyes for the system. While the quadrotor is in flight, the camera captures video of the ground in search of a designated target. The main processor in the processor-and-controller subsystem handles the image captured; when the target is in the camera's field of view, the processor calls for the location of the target from the flight controller and transmits it to the GCS. The system then adjusts its trajectory to hover at a vantage point where the target is at the center of the camera's view. On command, the laser designator fires a continuous electromagnetic pulse at the target to mark it for a precision strike. Throughout the process, the system stores the imagery captured in digital memory to allow future review of the engagement as required.

4. The Communication Subsystem

The communication subsystem consists of long-range and short-range transceivers. The long-range transceiver is responsible for establishing communication links with the GCS via radio frequency. The short-range transceiver is a critical component in enabling communications among a collaborative swarm of quadrotors executing the same mission. Either transceivers perform transmission on secured channels, with the data encrypted via the on-board the encryption key.

5. The Positioning Subsystem

The positioning subsystem consists of a GPS receiver and inertial-navigation system (INS). The GPS receiver attains the quadrotor's position coordinates from the GPS satellites via radio frequency. The INS measures the current position of the quadrotor and acts as a redundant system to GPS where there is a loss of GPS signals. The subsystem initializes the INS position solution during system boot up, and the GPS receiver updates the solution periodically. The two components work in tandem to provide accurate position coordinates to the flight controller. These coordinates ensure that the quadrotor follows the optimal trajectory generated for the system.

6. The Collision-Avoidance Subsystem

The collision avoidance subsystem consists of the LIDAR sensor now. Commanded by the main processor, the LIDAR applies radar principles and uses laser light to detect obstacles in the path of the quadrotor. It then transmits information of the heading and distance of the obstacles to the flight controller via the main processor, upon which the flight controller regenerates an optimal flight path to overcome the obstacle.

7. The Power-Supply Subsystem

The power supply subsystem consists of a battery, circuit breaker, and power-distribution box. The battery carries the electric charge necessary to power the entire system. The circuit breaker acts as a safety device by cutting electrical flow to the system, which may be necessary especially when human operators are in contact with components during initialization or maintenance. The power-distribution box distributes power from the battery to all other components in the system; critically, it controls the amount of power inputs supplied to the four rotors, based on the command output from the flight controller, as varying the power input alters the thrust produced at each rotor and in turn results in the different flight motions of the quadrotor.

H. A VERIFICATION TESTING SCENARIO

Based on the operational concept presented in Section II-D, the author developed a design scenario to conduct experiment for verification and validation of the feasibility for an autonomous target acquisition system using quadrotors. The thesis design scenario goes as follows:

Prior intelligence indicates that an adversary’s MLRS is within the vicinity of a sub-urban terrain. The division command echelon tasked an SAF STORM team to perform target acquisition before following up with an engagement strike to destroy the MLRS. Given the terrain, there are unlikely areas where MLRS will be. To optimize the search process, these impassable areas are pre-processed, blocked and excluded from search path generation.

Figure 5 shows a fictitious area of operation extracted from Google Earth beside the processed overlay. The left image is an aerial view of the entire area of operation. The right image is the processed output after blocking the improbable or impassable areas for a land target, which in this case, are lakes.

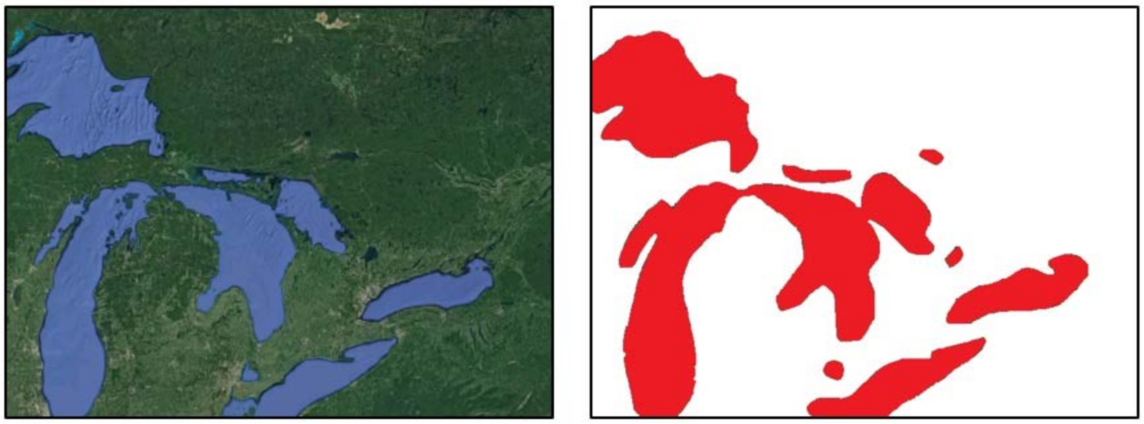

Figure 5. Area of Operation with Processed Overlay. Adapted from Google Maps (2016).

Operators then deployed two autonomous quadrotors to search for the MLRS along their autonomously found trajectories. Upon issuing the launch command, the operator at the GCS lets the system run autonomously. The autonomous system is capable of re-optimizing its initial planned trajectory in real-time should there be any obstacle in their path, and thereafter to maneuver around the obstacle. Upon positive acquisition of the target by the system, the quadrotor alerts the GCS operator, then loiters continuously to track the target.

THIS PAGE INTENTIONALLY LEFT BLANK

III. A MODELLING AND SIMULATION ENVIRONMENT

This chapter describes an experimental setup used to support the design scenario given in the previous chapter, addressing both hardware and software components.

A. THE AUTONOMOUS SYSTEMS ENGINEERING AND INTEGRATION LABORATORY

The Autonomous Systems Engineering and Integration Laboratory (ASEIL) is one of two indoor laboratories at Naval Postgraduate School allowing testing of quadrotors. Using equipment provided by Quanser, Inc., the laboratory provides a safe and controlled environment for the development of autonomous aerial and ground systems, employing Qball-X4 quadrotors, an indoor localization system and a desktop personal computer that acts as a ground control station.

The space available for indoor flight operations measures approximately 4.5m long x 4m wide x 2m high, as illustrated in Figure 6. Foam mats overlay the floor to cushion the landing or impact that the quadrotors may make. There are 12 infrared cameras, which line the perimeter of the flight-operation area as part of the indoor localization system. The figure shows them as yellow triangles. The GCS is located to the south of the experimentation area and the origin for the indoor localization system is marked with a red cross, as shown.

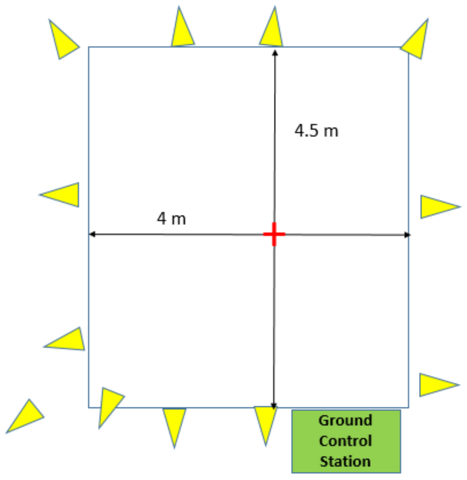

Figure 6. ASEIL Layout

Throughout the conduct of flight experimentation, the safety of operators and other non-involved personnel is a high priority. A safety net surrounds the flight-operation area to prevent quadrotor collisions with personnel and a warning on the exterior door indicates when a flight test is underway.

B. UAV HARDWARE

1. Qball-X4 Quadrotors

The QBall-X4 quadrotor ("QBall" for short) is a commercial product of Quanser, Inc. It is a rotary-wing vehicle platform designed for researchers investigating potential applications of UAVs, with an open developmental architecture. Its guidance and control algorithms are easily adjustable by modifying the baseline control scheme at the GCS. Figure 7 shows a graphic overview of the Qball.

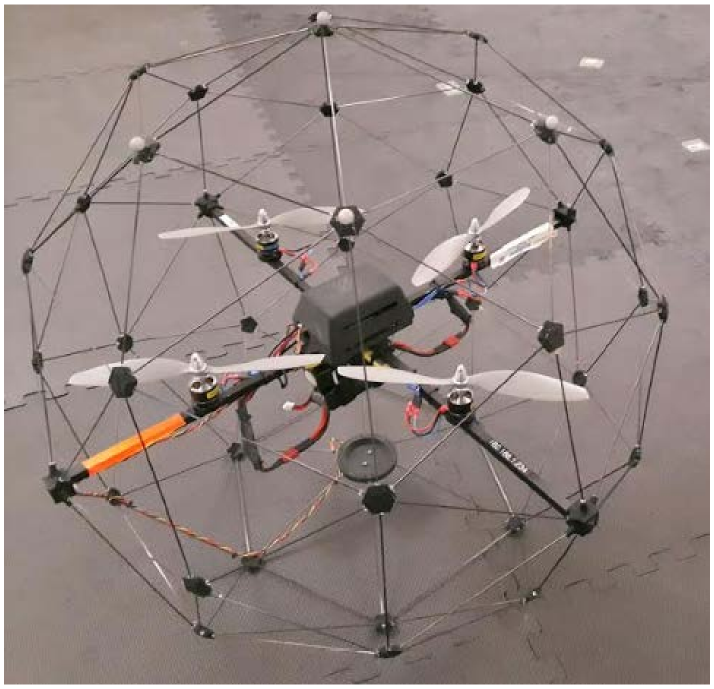

Figure 7. Qball-X4 Quadrotor

a. Motors and Propellers

Following a quadrotor helicopter design, the Qball has four 740KV motors fitted with paired counter-rotating 10″x4.7″ propellers (10 being the diameter of the propeller measured from blade tip to blade tip and 4.7 the pitch of the blade). Each motor connects itself to its own speed controllers. The set of motor, propeller and speed controllers collectively form a rotor. Each rotor is mounted equidistance from each other along the Z-plane on the “+” frame of the quadrotor. Cowling et al. recommended that the configuration be such that the front and back rotors spin clockwise, while the left and right rotors spin counter-clockwise (2010). This paired-opposite arrangement prevents the yaw of the Qball at equal rotating speed of all four rotors.

b. Protective Cage

The QBall is enclosed entirely within a lightweight carbon-fiber cage to protect the unit from the minor collisions common in an indoor laboratory without incurring a severe weight penalty. Resembling a Buckminsterfullerene, the structure also provides mounting points for reflective markers. The later section on the indoor localization system shall explain further on the purpose of these markers.

c. HiQ Data Acquisition Card (DAQ)/ Gumstix Embedded Computer

The HiQ DAQ is an on-board avionics data-acquisition board for the Qball. Integrated with a Linux-driven Gumstix embedded computer, HiQ controls the Qball by reading the data provided by on-board sensors and outputting commands to the motors through the motor-speed controllers. As specified by Quanser, the Input/ Output of the HiQ includes (2010):

- 10 pulse-width modulation (PWM) outputs (servo motor outputs)
- a three-axis gyroscope with range configurable for ±75˚/s, ±150˚/s, or ±300˚/s and resolution of 0.0125˚/s/LSB at a range setting of ±75˚/s
- a three-axis accelerometer with a resolution 3.33 mg/LSB
- six analog inputs, 12-bit, +3.3V
- a three-axis magnetometer, 0.5 mGa/LSB

- an eight-channel RF receiver input (optional)
- four Maxbotix sonar inputs
- two pressure sensors (absolute and relative pressure)
- input power of 10–20V

d. Batteries

Two three-cell 2500mAh LiPo batteries power the QBall, with each supplying a nominal voltage of 12.6V. The batteries are stacked vertically and strapped below the center of the quadrotor frame to maintain the QBall's center of gravity. After a full charge, the batteries allow approximately 10 minutes of flight time.

e. Joystick

The user can fly the Qball manually, and control inputs are via the joystick provided by Quanser. Connected to the GCS through a universal serial bus (USB) port, the joystick has two components that translate lever motion into throttle, yaw, pitch and roll commands, as shown in Figure 8. Four sliders are available to supply trim signals for the four flight controls. Significantly, the joystick provides activation and kill switches for the QBall during autonomous operation. The Qball takes off upon receiving a small gain in throttle command and ceases flight when there is zero throttle gain.

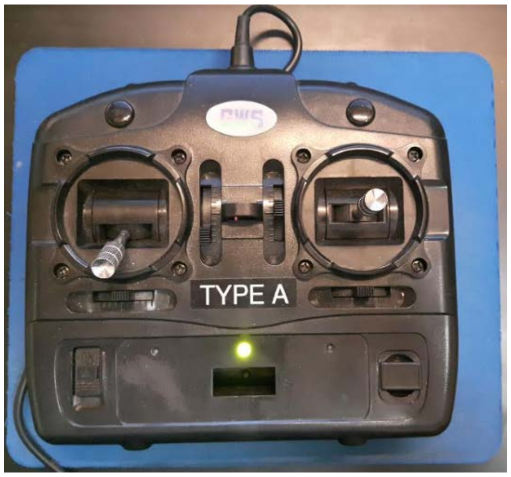

Figure 8. Controller for Manual Operations

2. The OptiTrack Motion-Capture System

The OptiTrack Motion Capture System is an indoor camera-based localization system developed by NaturalPoint. OptiTrack uses the installed Flex 3 infrared cameras to detect the reflective markers on the Qball's protective cage. This allows the OptiTrack system to capture position and orientation throughout its flight. Table 2 lists the specifications of the infrared camera.

Table 2. Specifications of OptiTrack Infrared Camera. Source: OptiTrack (2014).

Capability	**Specification**
Horizontal Field of View	46.2 degrees
Vertical Field of View	34.7 degrees
Resolution	640 x 480 pixel
Frame Rate	25,50, 100 frames per second

These cameras mounted around the walls of the ASEIL, near the ceiling, maximizes the capture volume of the laboratory as shown in Figure 9. The dimensions of the capture volume are approximately 4.5 m in length, 4 m in width and 2m in height, reflecting the available indoor flight-experimentation area. As long as the Qball remains within the capture volume, the system can perform six degrees of freedom (6DOF) tracking of the Qball.

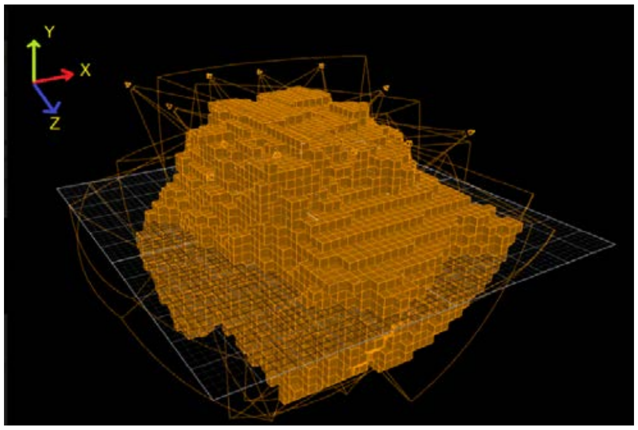

Figure 9. Capture Volume within ASEIL

C. SOFTWARE ARCHITECTURE

1. MATLAB and Simulink

This thesis used MATLAB and Simulink extensively in the development of the control algorithms. The model-based Simulink software allows rapid configuration without tedious coding.

2. QUARC Real-Time Control Software

Quanser's QuaRC real-time control software, a rapid control prototyping product, is the default software for controlling the QBall. Its architecture builds on the MATLAB and Simulink environments to facilitate the design and development of real-time applications. Quanser states that "QuaRC integrates seamlessly with Simulink's high-level graphical environment and allows Simulink diagrams to interface hardware and be run in real-time on a variety of local and/or remote targets" (2016, 2).

Embedded within the software is a communication tool that bridges the GCS and the QBall's Gumstix computer. Transmission Control Protocol (TCP) and Internet Protocol (IP) establish the connection between the two hardware.

3. The OptiTrack Tracking Tool

The OptiTrack application serves as the software interface for the OptiTrack Motion Capture System. The user needs to perform two critical procedures through the application for accurate tracking by the indoor localization system, as follows.

a. Camera Calibration

Calibration of the cameras is required to orientate the system and form the capture volume. Using a proprietary wand as shown in Figure 10, "wanding" or moving the wand in a figure-of-eight pattern around the indoor environment is carried out while the cameras tracks the three IR markers at the wand's tip (seen as illuminated balls in the figure).

Figure 11 shows the output of the wanding. The 12 boxes on the left of the figure show traces of the reflective markers captured by the 12 IR cameras. The block at the right of the figure indicates the number of data points captured by each camera, and alerts the user if the system captured sufficient data points.

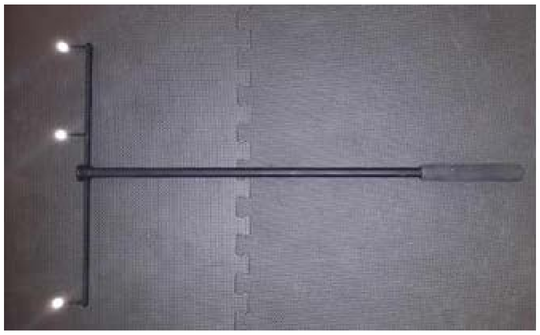

Figure 10. Wand Provided by NaturalPoint to Enable Camera Calibration

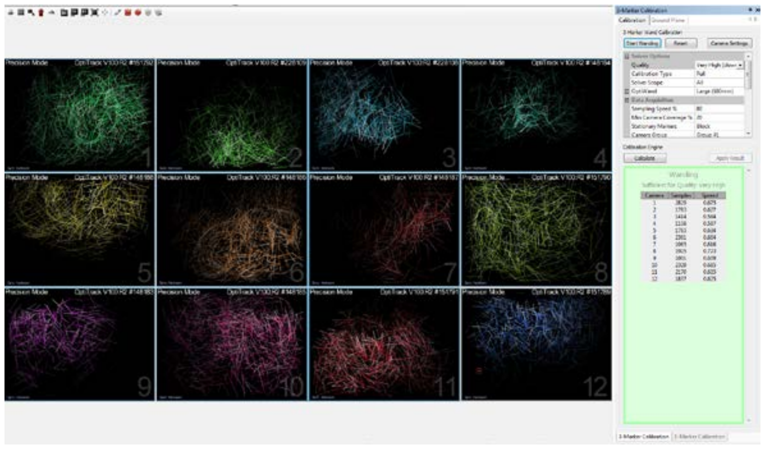

Figure 11. Output of Wanding

Upon receiving sufficient data points for high quality calibration, the software prompts the user to start the calibration process. The software computes and calibrates the cameras based on these data points. Thereafter, the user set the ground plane by placing a calibration square, as shown in Figure 12, at the desired point of origin. The localization system then picks up the coordinates of the reflective markers. When setting the calibration square on the ground plane, the longer arm points to the ground control station, indicating to the localization system the direction of positive Z-axis. By default, the right-hand coordinate system is applied with the Z-axis is pointing towards to the ground control station.

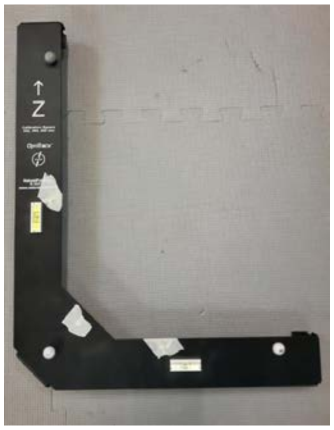

Figure 12. The 50mm Calibration Square Provided by NaturalPoint

b. Calibration of Qball

The OptiTrack system names objects for tracking as “trackables.” There is a need to calibrate the trackables for the OptiTrack system to detect, recognize and track them. The user first need to affix the object with the IR markers. If the user needs to track more than one objects at the same time, the arrangement of the IR markers has to be unique to allow the system to distinguish between the objects.

The Qball is the object of interest and the author mounted four IR markers on its protective cage in an asymmetrical manner along all axes so that the OptiTrack system can detect its orientation. Figure 13 illustrates how the Qball appears in the motion tracking software through the reflective IR markers. The four nodes circled in red correspond to the IR markers mounted on the Qball. The image at the top left shows a top view (Y-plane) of the nodes. The Z-axis is pointing to the bottom of the figure and the X-axis to the right. The image at top right presents a side view of the nodes down the negative Z-axis. The image at bottom is a perspective view.

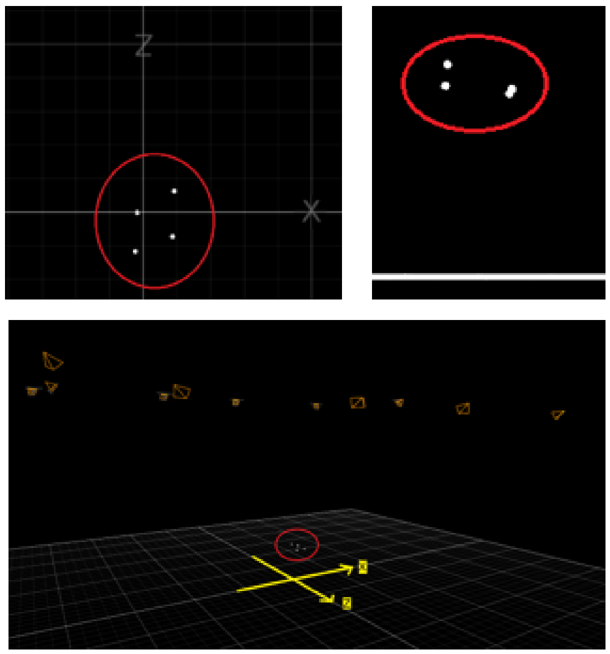

Figure 13. Qball as Seen from the Perspective of the OptiTrack System

D. QUADROTOR MODELING

The Qball is a dynamic model that can represents most of the commercially available Quadrotors. Before being able to manipulate the control algorithms of the Qball, it is necessary to understand the flight control actuation of quadrotor.

1. Defining the Coordinate Frames

The coordinate frame defines the directions of the Cartesian axes and in turn, the Euler angles for flight control. In the case of the Qball, the author applied a body fixed coordinate frame centered at the center of mass of the Qball, as seen in Figure 14a. The X-axis is pointing to the front of the Qball, and a rotation about the X-axis creates rolling moment. The Y-axis is pointing to the left of the Qball, and rotation about the Y-axis creates a pitching moment. Finally, the Z-axis points above the Qball, and rotation about the Z-axis creates a yawing moment. All axes follow the right hand rule.

Now, it is appropriate to emphasize that the Qball axes differs from the coordinate system of the OptiTrack System. From the perspective of the indoor localization system, the axes differ in direction as reflected in Figure 14b. The X-axis points positively to the right of the Qball, the Y-axis points positively upwards and the Z-axis points positively to the rear of the Qball in the direction of the GCS. Hence, there is a need for the user to translate the OptiTrack coordinates, which acts as the global coordinates, accordingly to the Qball's local frame of reference for the Qball to move as per the desired direction.

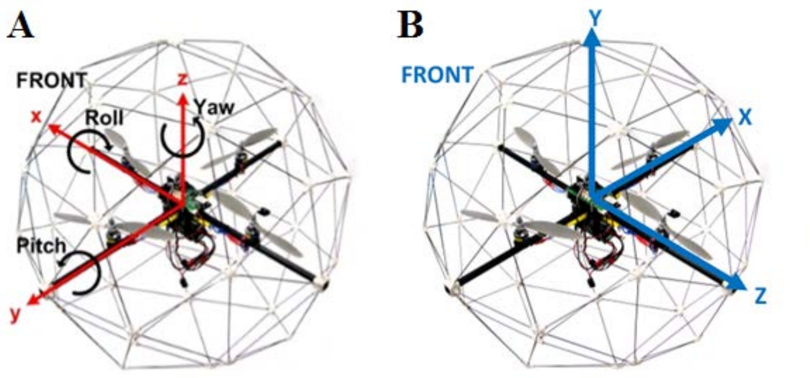

Figure 14. Qball-X4 Axes and Sign Conventions. Adapted from Quanser (2010).

2. Dynamics of Quadrotor Actuators

The thrust generated by each of the mounted propellers determines the elevation, pitch, roll and yaw of the Qball. Unlike a fixed-wing airplane whereby rudder and ailerons control the attitude of the airplane, a quadrotor accomplishes these motions by varying the speed at each rotor. Therefore, it is pertinent to understand the dynamics of the quadrotor actuators to gain control over the quadrotor.

a. Thrust

The thrust F that is generated by the individual rotors follows the first-order system seen in equation (1), where u is the pulse-width modulation (PWM) input to the actuators, ω is the actuator bandwidth and K is a positive gain. Therefore, by varying the amount of PWM input supplied to each of the motors, the system can individually control the amount of thrust generated by each propeller.

$$F = K \frac{\omega}{s+\omega} u \qquad (1)$$

b. Roll and Pitch

Speed variation in each of the propellers produces force vectors that translate to the pitch, roll and yaw of the Qball. There is positive roll when the left rotor spins faster than the right rotor. There is positive pitch when the front rotor has a lower PWM input and spins slower than the rear rotor.

Equation (2) calculates the Euler angle in the X-axis, or roll angle. *J* is the rotational inertia of the Qball in the roll axes; *L* is the distance between the center of the propeller and the Qball's center of gravity; and *ΔF* is the net force generated between the two motors involved in the pitch/roll motion. The same equation can determine the pitch angle θ.

$$J\ddot{\phi} = \Delta F L \qquad (2)$$

c. Yaw

There is yaw about the Z-axis when there is a resultant torque between the two clockwise rotating propellers (the front and back rotors) and the two counter-clockwise rotating propellers (the left and right rotors). Figure 15 illustrates a model of the yaw axis with the directions of the propeller rotation shown. The Z-axis of the body-fixed coordinate frame points out of the page; and hence, positive yaw is an anti-clockwise moment. Equation (3) calculates the Euler angle about the Z-axis or yaw angle, where the resultant torque is $\Delta\tau = \tau_1 + \tau_2 + \tau_3 + \tau_4$.

$$J_{yaw}\ddot{\psi}_y = \Delta\tau \qquad (3)$$

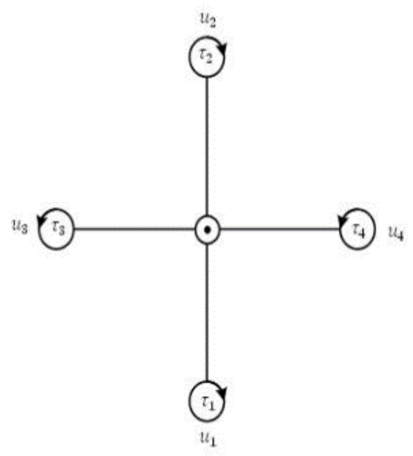

Figure 15. Yaw Model of Qball

d. Height

How much thrust all four propellers provide determines the height of the Qball. Equation (4) represents the dynamic model of the Qball's height, where *m* is the total mass of the Qball, *h* is the height attained, *F* is the thrust generated by each propeller, ϕ is the roll angle, θ is the pitch angle, and *g* is the acceleration due to gravity.

$$m\ddot{h} = 4F\cos\phi\cos\theta - mg \tag{4}$$

The overall thrust vector is a function of the pitch and roll angle, and therefore the Qball will not be perpendicular to the ground if any angle is non-zero.

e. System Parameters

Table 3 summarized the values of the various system parameters in the given dynamics equations.

Table 3. System Parameters of Qball-X4. Source: Quanser (2010).

Parameter	Value
K	120 N
ω	15 rad/sec
$J = J_{roll} = J_{pitch}$	0.03 kgm^2
J_{yaw}	0.04 kgm^2
m	1.4 kg
L	0.2 m
K_y	4 Nm

f. State and Control Vectors

With actuated dynamics reviewed, it is here appropriate to express the total normalized thrust and the second-order derivatives of the Euler angles in a control vector $\hat{u}$ as formulated in equation (5), where $u_1 = \tau_1 + \tau_2 + \tau_3 + \tau_4$, $u_2 = \ddot{\phi}$, $u_3 = \ddot{\theta}$ and $u_4 = \ddot{\psi}$.

$$\hat{u} = \left[u_1 + u_2 + u_3 + u_4\right]^T \tag{5}$$

The position coordinates and Euler angles can be expressed in twelve-state vector as formulated in equation (6), where $\left[x, y, z\right]^T$ is the translational position of the Qball center of gravity in the North-East-Down coordinate frame and $\left[\phi, \theta, \psi\right]^T$ is the attitude vector that contains the three Euler angles (Cowling et al. 2010).

$$x = \left[x, y, z, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}\right]^T \tag{6}$$

Typically, the equations of motion of aerial vehicles use the standard aeronautical rotational matrix R_{xyz}. Cowling et al., however, has simplified the equations of motion of the quadrotor using an alternative rotational matrix R_{zyx} (2006). Applying the rotational matrix to the body-fixed coordinate frame of the Qball, the translational equations of motion become the expressions shown in equations (7), (8) and (9):

$$\ddot{x} = -u_1 \cos\phi \sin\theta \tag{7}$$

$$\ddot{y} = -u_1 \sin\phi \tag{8}$$

$$\ddot{z} = u_1 \cos\phi \cos\theta - g \tag{9}$$

With the preceding set of equations, the first derivative of the state vector is in the form shown in equation (10).

$$\dot{x} = \frac{d}{dt}\begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \phi \\ \theta \\ \psi \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ -u_1 \cos\phi \sin\theta \\ -u_1 \sin\phi \\ u_1 \cos\phi \cos\theta - g \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{10}$$

To address the subsequent chapter of trajectory generation using direct method, Cowling et al., (2006) proposed the desired output of the system y to contain the translational positions and yaw angle as seen in equation (11).

$$y = \begin{bmatrix} x \\ y \\ z \\ \psi \end{bmatrix} = \begin{bmatrix} I_{3x3} & 0_{3x5} & 0 & 0_{3x3} \\ 0 & 0_{1x5} & 1 & 0_{1x3} \end{bmatrix} x = Cx \tag{11}$$

IV. PLANNING AND OPTIMIZATION OF THE SEARCH PATH

Obviously, any deliberate action begins with a plan. This applies to an autonomous system, which follows and executes the plan of a commanding operator. For a target-acquisition system, target identity and location are key. While autonomous planning algorithms and high-level processes for developing artificial intelligence (AI) are beyond the scope of this research, the author has adopted the principle of developing a logical sequence of actions "to transform an initial world state into a desired goal state" (LaValle 2006, xi). LaValle cites several factors that are basic to any plan:

- State space
- Time
- Criterion
- Set of Actions

A. SEARCH-PATH PLANNING

1. State Space

The area of search operations defines the initial state space of the plan. Starting from the location of the GCS, an autonomous system may start searching for the target from any direction. The current state space may be vast, and it is not necessarily logical to search the entire area—for example, a land target is highly unlikely to be over a body of water. The user can eliminate such implausible area from the initial state space. With a smaller state space, lesser computational resources are required to find the optimized path.

In contrast to an AI system, defining and even reducing the state space is relatively simpler for an autonomous system. The reason is that the system does not entirely exclude presence of man, and that he can performs this role to understand the world on behalf of the autonomous system. At the start of the operation when planning takes place, the man-in-the-loop is able to decipher the start state and input only the reduced conditions to the autonomous system. Thus, the autonomous system has a smaller state space to work with and can produce faster results.

2. Time

A plan can either explicitly model or implicitly represent time. In this optimization of a search path, both expressions of time are important and critical. Firstly, explicit time is a constraint to an optimization process. The objective for the search path optimization is essentially to cover the entire plausible area in the shortest amount of time by travelling the least amount of distance. As for implicit time, it is a reflection of having to follow the sequence of waypoints generated by the optimization process. Order matters to ensure optimality.

3. Criterion

LaValle shares that there are generally two types of criteria, which influences the planning outcome, namely feasibility and optimality (2006). According to LaValle, feasibility disregards efficiency and search for a plan that achieve the goal state; whereas optimality searches for a feasible plan while satisfying constraints imposed on the system. Obviously, the search-path optimization falls into the second criterion.

Two constraints must be satisfied to qualify as an optimal path: (a) the path has to be of the shortest in length, which translate to shortest time to completion; and (b) economy of traversal—the path has to be completed in a single loop without revisiting the same area.

4. Set of Actions

With state space, time and criterion defined, the plan now must specify the "actions that manipulate the state" (LaValle 2006, 18).

As mentioned earlier on the role of the man in the loop, the operator is responsible for, and highly capable of processing an area of operation quickly to eliminate implausible search area. He then passes on this useful information to the autonomous system. With the boundaries of the area of operation defined, the system's processor can commerce searching for an optimized path by first discretizing the plausible areas into nodes. This action facilitates the computational work to satisfy the constraints.

Next, the constraints are set as discussed earlier. The processor computes the distances between nodes and finds the shortest route to connect all nodes in a single loop.

Finally, the operator activates the autonomous system to begin searching based on the optimal search path generated.

B. SEARCH PATH OPTIMIZATION

The author has deliberately modelled the existing problem after the travelling salesman problem (TSP). The latter is a classical non-deterministic polynomial-time (NP)-hard problem, which has numerous known heuristics and exact algorithms. Just as the TSP looks for the shortest possible route that visits all the cities exactly once before returning to the city of origin, a target-acquisition search path attempts to find the shortest possible path to cover all nodes exactly once before returning to the originating GCS.

The most straightforward avenue towards solving the TSP is to use linear programming to work out all possible combinations of the solution, compute the associated costs of each combination, and determine the combination with the least cost. Such a brute-force solution is practical only for a small number of nodes, however, because the number of combinations grows exponentially. Nevertheless, the solution has the virtue of exactitude.

The thesis formulates the optimized search path problem as a binary integer linear program (ILP). The author then uses MATLAB to generate all possible connections between two nodes and compute the respective straight-line distance. The objective function for the ILP is to minimize the total distance required to connect all nodes, as expressed in Equation (12). Equation (13) defines the binary decision variable, where "1" bit represents a chosen connection and "0" bit represents otherwise. A series of inequalities serves as constraints to restrict the solution. Equations (14) and (15) ensure that there is exactly one arrival and departure to and from each node respectively. Equation (16) enforces the feasible solution to be of a single tour, which is one continuous loop connecting all nodes. These equations are given as follows:

$$\min \sum_{i=0}^{n} \sum_{j \neq i,\, j=0}^{n} d_{ij} x_{ij} \tag{12}$$

Here d_{ij} is the distance between nodes i and j, and x_{ij} is the decision variable to determine whether nodes i and j are connected to each other.

$$0 \leq x_{ij} \leq 1, x_{ij} \in \square \tag{13}$$

$$\sum_{i \neq j, i=0}^{n} x_{ij} = 1 \tag{14}$$

$$\sum_{j \neq i,\, j=0}^{n} x_{ij} = 1 \tag{15}$$

$$x_i - x_j + n x_{ij} \leq n - 1 \tag{16}$$

C. SIMULATION SCENARIO

To demonstrate the feasibility of using ILP to optimize the search path, the thesis performs a simulation of the possible planning operation. The code used in captured in Appendix B.

1. Pre-processing

As shown in Figure 16, pre-processing starts with creating an overlay on the map containing the area of search (see left image). Then, the areas that are impassable for the target are shaded on the overlay (see middle image). The author then saved the overlay in joint photographic experts group (JPEG) format and imported it into MATLAB environment as a three-layer array. The author carried on to flatten the array to a single layer to speed up its processing. To further increase the speed, the author converted the single layer array into a binary array (see right image). The shaded area is represented by “0” while the empty area is represented by “1.” The reader can safely assume that the imported image is undistorted.

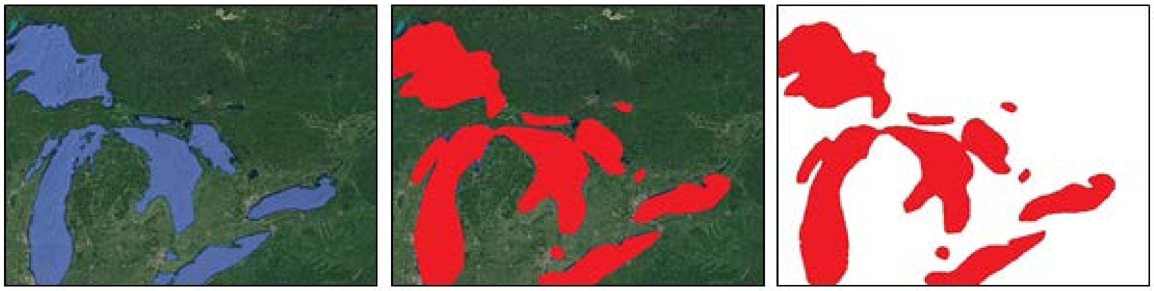

Figure 16. Pre-Processing of Area of Operation

Based on the size of the image, the code creates two vectors to represent the X- and Y coordinates of the map. These vectors form the basis for the location of the nodes. In an image with a resolution of 640 x 480, each of these vectors has 307200 elements, one for each pixel. The element locations of the shaded area in the binary matrix are then determined to remove the elements of the same position in the two coordinate vectors. Now, the reader can see the vectors as “to search” vectors.

Given that the search camera has a finite field of view, the user can reduce the “to search” vector by aggregating those coordinates that can be “seen” by the camera as it flies over a single point. The code only considers the width of the camera’s field of view for ease of computation. Figure 17 shows the interim output, which depicts the nodes and the shaded area.

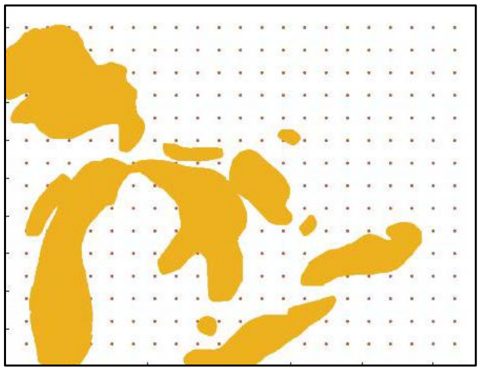

Figure 17. Possible Detection Area Discretized with Nodes

2. Optimization

The author implemented the ILP, as discussed above, in MATLAB environment. Using the MATLAB in-built optimization solver, a single loop connecting all the nodes generated shown in the Figure 17 forms the optimized search path. Figure 18 shows the path as dotted line on the map. Based on a camera with its width of vision at 30m, 236 equidistance nodes were plotted in non-shaded area on an otherwise 336 nodes in the entire area of operation of dimension 640m x 480m. The total distance covered in an optimized path is approximately 7.25km instead of approximately 9.80km when combing through the entire area of operation.

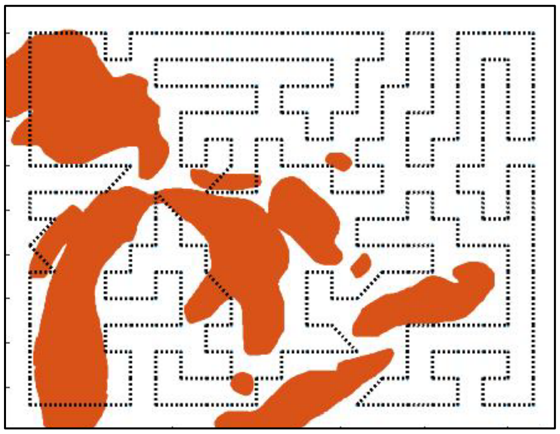

Figure 18. Optimized Search Path

The essential output of the optimization run is the sequence that connects the nodes in a single tour and the respective coordinates. This forms the optimal waypoints for the Qball's traversal.

3. Obstacles Avoidance

The search for an optimized search path has assumed a flat terrain with no major impediments. The assumption is not true in reality. Hence, there will be a need to execute certain maneuver to overcome the obstacles in the path of the system. The system can fly either around or over the obstacles. The next chapter shall go on to present the methodology of generating the trajectory necessary to overcome the obstacles.

V. DIRECT METHOD-BASED TRAJECTORY GENERATOR

Given the finite processing ability of an autonomous vehicle and the need for rapid response in a dynamic environment, it is imperative that an autonomous processor be capable of making decisions on its trajectory from limited input. The direct method of calculus of variations avails as a potential methodology of choice.

There is two part to the direct method:

First, by exploiting the inverse dynamics of the Qball, optimization can occur in the output space instead of the control space, because inverse dynamics expresses the state and control vectors as functions of the output. Moreover, this results in differentially flat derivatives.

Next, the method performs optimization in the virtual domain instead of the conventional time domain. This decouples the time and space parameterizations, whose combination tends to create complexity in resolving the variables. The applied methodology is therefore inverse dynamics in the virtual domain or IDVD method.

The central idea of IDVD is to use only a small set of variables to achieve a near-optimal solution. Because lesser parameters are required for computation, the computational time to determine a quasi-optimal trajectory is significantly less than with other direct methods.

A. CONTROLLER ARCHITECTURE

Cowling et al. have proposed a controller architecture that is suitable for trajectory generation (2010), as illustrated in Figure 19. Two major entities form the architecture seen below, which are the trajectory generator and trajectory follower. They are separated by the dotted line in the figure, with the trajectory follower occupying the upper half.

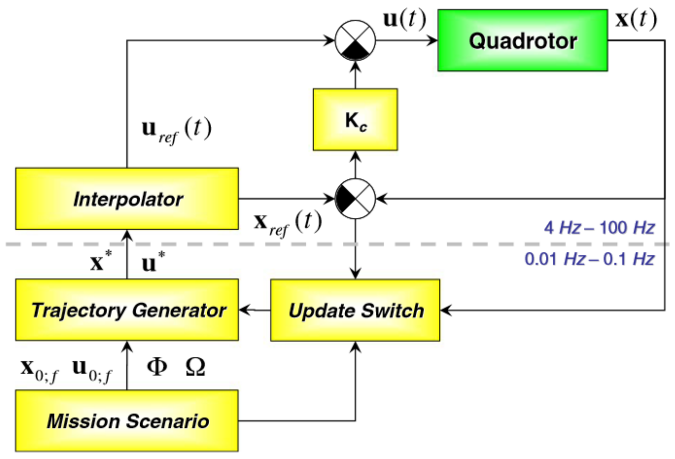

Figure 19. General Architecture of Controller. Source: Cowling et al. (2010).

The role of the trajectory generator is to produce a feasible near-optimal path for the Qball depending on the operating scenario. The system generates the trajectory with consideration of boundary (initial and final) conditions, mission-performance index and other flight constraints. While the solution is feasible and near optimal, it may not necessarily be the most optimal. Nevertheless, given the relatively short time needed to generate the trajectory, it remains advantageous to adopt the direct method in the controller as befitting a time-sensitive environment.

The trajectory generator is also capable of generating new trajectories in real-time, which enables re-optimization during flight. In the context of an autonomous target-acquisition system in the military domain, there may be new situation updates such as new threats found in the area of operation. This would require alterations in the flight path of the quadrotor. Such an ability to redirect the quadrotor is critical to the mission commander. In addition, there tends to be some discrepancy between the actual path taken and the trajectory generated, owing to disturbances or imperfections in the controller. Hence, the presence of an update switch in the architecture that enables updating of the trajectory generator with a new set of conditions however big or small the changes are. Cowling et al. recommended the updates to run at a frequency of 0.01 Hz to 0.1Hz to maintain optimality (2010). This is, however, dependent on the on-board processing ability of the quadrotor.

Trajectory following calls for the presence of an interpolator to ensure that the quadrotor maintains the reference trajectory by detecting disturbances that might cause deviation. Hence, the architecture uses linear quadratic regulator (LQR) to track and monitor the real-time trajectory (Cowling et al. 2006). The LQR runs at a higher frequency of 4 Hz–100 Hz to detect and counter disturbances to the reference trajectory.

B. INVERSE DYNAMICS IN THE VIRTUAL DOMAIN

According to Cowling et al., the reader can see the inverse dynamic in the virtual domain (IDVD) as a four-part process (2010). First is the generation of a reference trajectory that is independent of the time domain. Second is the use of a speed factor to revert the time-independent reference trajectory from the virtual domain to back to time domain. Third is to utilize inverse dynamics to determine the controls necessary for the Qball's motion. Fourth and finally, is the optimization of the trajectory, which is constrained by the boundary conditions.

1. The Reference Trajectory

To decouple space and time, Cowling et al. suggest the use of a virtual arc τ to express trajectory to enable optimization of the velocity history independently along the trajectory (2010). The virtual variable serves as a reference function in the virtual domain and varies between zero and a finite value τ_f.

The IDVD method needs parameterization of the trajectory. Many different parameterizations can approximate the three Cartesian coordinates that represent the trajectory. All methods bear similarity to Equation (17) which involves the product of a free variable a_k and a basis function Γ_f.

$$P(t) = \sum_{k=0}^{M} a_k \Gamma_f (t) \tag{17}$$

The order of parameterization, M, is defined by the number of boundary conditions that needs to be satisfied. It is formulated in equation (18), where d_0 is the highest-order spatial derivative of the initial conditions and d_f is likewise of the final conditions. The reader can further increase this order if a higher degree of freedom is desired.

$$M = d_0 + d_f + 1 \tag{18}$$

According to Chua, the trajectory can hence be parameterized using trigonometric terms (2013). The X-, Y- and Z-coordinates are formulated as shown in equations (19), (20) and (21), where $\overline{\tau} = \frac{\tau}{\tau_f}$. Given that both the initial and final conditions have their highest-order of derivatives as three, the order of parameterization would be seven in this case.

$$x\left(\overline{\tau}\right) = P_x\left(\overline{\tau}\right) = a_{x0} + \sum_{i=1}^{3} a_{xi}\cos\left(i\pi\overline{\tau}\right) + \sum_{i=1}^{4} b_{xi}\sin\left(i\pi\overline{\tau}\right) \tag{19}$$

$$y\left(\overline{\tau}\right) = P_y\left(\overline{\tau}\right) = a_{y0} + \sum_{i=1}^{3} a_{yi}\cos\left(i\pi\overline{\tau}\right) + \sum_{i=1}^{4} b_{yi}\sin\left(i\pi\overline{\tau}\right) \tag{20}$$

and

$$z\left(\overline{\tau}\right) = P_z\left(\overline{\tau}\right) = a_{z0} + \sum_{i=1}^{3} a_{zi}\cos\left(i\pi\overline{\tau}\right) + \sum_{i=1}^{4} b_{zi}\sin\left(i\pi\overline{\tau}\right) \tag{21}$$

To determine the unknown coefficients, Chua suggested to resolve the following system of equations shown in (22) for each of the three Cartesian coordinates. For the right-hand side of the equation, x_0 and x_f represents the initial and final states of the Qball. Together with the first and second-order derivatives $\left(x_0^{'}, x_f^{'}, x_0^{''}, x_f^{''}\right)$, they form the constraints to be satisfied. The third-order derivatives $\left(x_0^{'''}, x_f^{'''}\right)$, also known as the initial and final jerk, remains as free variables.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \pi & 2\pi & 3\pi & 4\pi \\ 0 & 0 & 0 & 0 & -\pi & 2\pi & -3\pi & 4\pi \\ 0 & -\pi^2 & -(2\pi)^2 & -(3\pi)^2 & 0 & 0 & 0 & 0 \\ 0 & \pi^2 & -(2\pi)^2 & (3\pi)^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\pi^3 & -(2\pi)^3 & -(3\pi)^3 & -(4\pi)^3 \\ 0 & 0 & 0 & 0 & \pi^3 & -(2\pi)^3 & (3\pi)^3 & -(4\pi)^3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} x_0 \\ x_f \\ x_0^{'}\tau_f \\ x_f^{'}\tau_f \\ x_0^{''}\tau_f^2 \\ x_f^{''}\tau_f^2 \\ x_0^{'''}\tau_f^3 \\ x_f^{'''}\tau_f^3 \end{bmatrix} \quad (22)$$

2. Speed Factor

To optimize the trajectory in the output space, space and time were decoupled using a virtual arc, τ. To switch from the virtual domain back to the time domain for controlling the quadrotor, or in another words to separate the trajectory from the speed profile a variable speed factor that relates virtual arc τ to time as shown in equation (23) is employed.

$$\lambda(\tau) = \frac{d\tau}{dt} \qquad (23)$$

The reader can change the speed profile along the reference trajectory by varying the speed factor as shown in equation (24).

$$V(\tau) = \lambda(\tau)\sqrt{x^{'}(\tau)^2 + y^{'}(\tau)^2 + z^{'}(\tau)^2} \qquad (24)$$

3. Differential Flatness

An important feature of the IDVD is the exploitation of the differential flatness property of the system, such that the author can express the system's states and controls using the output vector and its derivatives (Koo and Sastry 1999). Therefore, the author can represent the roll and pitch angles as a function of the output vector as shown in equations (25) and (26).

$$\theta = \arctan\left(\frac{\ddot{x}}{g - \ddot{z}}\right) \qquad (25)$$

and

$$\phi = \arcsin\left(\frac{-\ddot{y}}{\sqrt{\ddot{x}^2+\ddot{y}^2+(g-\ddot{z})^2}}\right) \tag{26}$$

Their respective first derivatives then become those shown in equation (27) and (28).

$$\dot{\theta} = \frac{x(g-\ddot{z})+\ddot{x}z}{(g-\ddot{z})^2+\ddot{x}^2} \tag{27}$$

and

$$\dot{\phi} = \frac{(\ddot{x}(\dddot{x})-(g-\ddot{z})\dddot{z})\ddot{y}-(\ddot{x}^2+(g-\ddot{z})^2)\dddot{y}}{(\ddot{x}^2+\ddot{y}^2+(g-\ddot{z})^2)\sqrt{\ddot{x}^2+(g-\ddot{z})^2}} \tag{28}$$

Singularity may occur during computation when $g-\ddot{z}$ or when the quadrotor is in free fall. The reader can avoid singularity by ensuring that the control vector is greater than zero and that both θ and ϕ are greater than 90°.

With the differentially flat equations above, optimization of the trajectory can occur in the output space, which decreases computational time tremendously as compared to using variables in the control space.

4. The Cost Function

Optimization involves achieving an outcome that fulfills an objective given certain constraints. To determine the optimality of the trajectory found, it necessitates qualitative measurement by means of a cost function. The cost function derived by Chua is a combination of the performance index of the trajectory and weighted penalties for not meeting the constraints, which are formulated in equation (29) and (30) respectively (2013).

$$PI = (1-w)\frac{1}{t_f}\int_0^{t_f}\sqrt{P_h(\dot{x}^2+\dot{y}^2)+P_v\dot{z}^2}\,dt + w(t_f - T)^2 \tag{29}$$

and

$$
\begin{aligned}
J = PI &+ w_1\left(\frac{(t_{desired} - t_{end})^2}{t_{desired}{}^2}\right) \\
&+ w_2\left(\begin{aligned}&\max\left(0, \frac{|\phi|_{\max} - \phi_{threshold}}{\phi_{threshold}}\right)^2 \\ &+ \max\left(0, \frac{|\theta|_{\max} - \theta_{threshold}}{\theta_{threshold}}\right)^2\end{aligned}\right) \\
&+ w_3\left(\max\left(0, \frac{d_{threshold,obs} - d_{\min,obs}}{d_{threshold,obs}}\right)^2\right) \\
&+ w_4\left(\begin{aligned}&\max\left(0, \frac{|X|_{\max} - X_{threshold}}{X_{threshold}}\right)^2 + \max\left(0, \frac{|Y|_{\max} - Y_{threshold}}{Y_{threshold}}\right)^2 + \\ &\max\left(0, \frac{|Z|_{\min} - Z_{\min,threshold}}{Z_{\min,threshold}}\right)^2 + \max\left(0, \frac{|Z|_{\max} - Z_{\max,threshold}}{Z_{\max,threshold}}\right)^2\end{aligned}\right)
\end{aligned}
\tag{30}
$$

The constraints included by Chua involve limitations due to arrival time at the final conditions, the QBall’s attitude, the presence of obstacles, and the spatial dimensions of the ASEIL.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. MIXING GUIDANCE COMMANDS

Building on the trajectory optimization presented in Chapter III, this chapter discusses how to leverage the existing control scheme developed by Quanser to rapidly prototype a feasible control architecture for an autonomous quadrotor.

A. THE QUANSER CONTROL SCHEME

The control scheme developed by Quanser allows the user to switch between manual joystick control and autonomous position and height control. The manual joystick control gives the operator full flight control of the Qball while the autonomous position and height control leverages on the position tracking input supplied by the OptiTrack Camera System to pilot the Qball autonomously. Quanser has built the entire control model in Simulink environment, which enables user to modify the model easily. Figure 20 illustrates the main page of Quanser's control model. It consists of 10 sub-subsystems to translate user input to commands executable by the Qball. Of interest to this thesis is the modification made to the "Position Commands" subsystem, which will be explained in a later section.

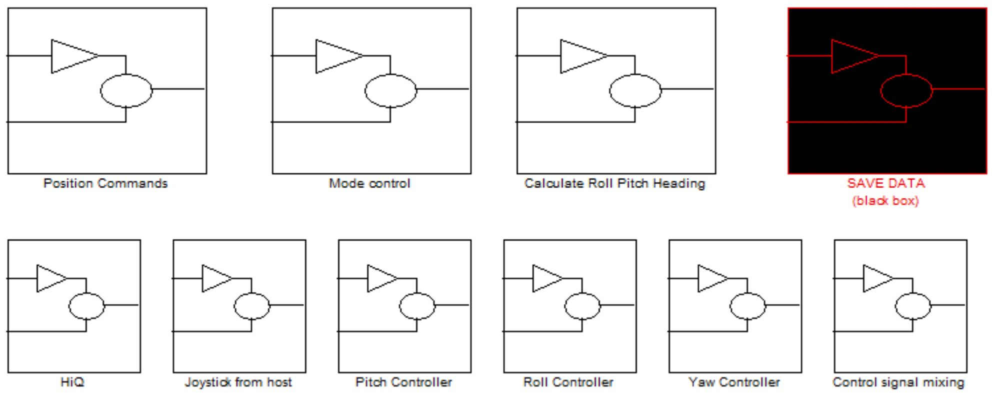

Figure 20. Quanser's Controller Representation in Simulink

The “mode control” subsystem allows the user to opt for joystick manual control or autonomous control via the “Position Commands” subsystem. The “Calculate Roll Pitch Heading” computes the roll, pitch and heading angles during the Qball’s flight using the input from the gyroscopes and accelerometers on the Qball. The “Joystick from host” subsystem is responsible to communicate with the joystick connected to the host station. It receives the various command input from the user via the joystick and directs them to the relevant subsystem controller. It also acts as the interface to receive coordinates input from the OptiTrack System. The latter is an important input in autonomous mode. The three controller subsystems—“Pitch Controller,” “Roll Controller” and “Yaw Controller,” processes the joystick commands or input from the OptiTrack System to produce control signals. The “Control Signal Mixing” block receives the signals from the controllers and translate them into motor output command. The “HiQ Subsystem” is the main interface block to communicate with the data acquisition card on the QBall. It reads the motor output from the “Control signal mixing” block and sends PWM signals to the Qball. These signals are responsible for commanding the actuators, which translate to motion of the Qball. Finally, the “Save Data” block outputs all the flight data into a binary MATLAB file, which facilitates analysis subsequently.

The original “Position Commands” subsystem, as shown in Figure 21, allows the user to use the slider gain block to generate heading, height and position commands for the Qball. These commands are more intuitive to the user as the frame of reference is in the coordinate or state space rather than control space when using the joystick. The user can toggle the sliders during mid-flight to command the Qball to move autonomously to the desired coordinates. This is a useful feature as only coordinates or waypoints are required to command the Qball. The subsequent sections in this chapter shall present the use of this feature in the control scheme to achieve autonomy in the Qball system.

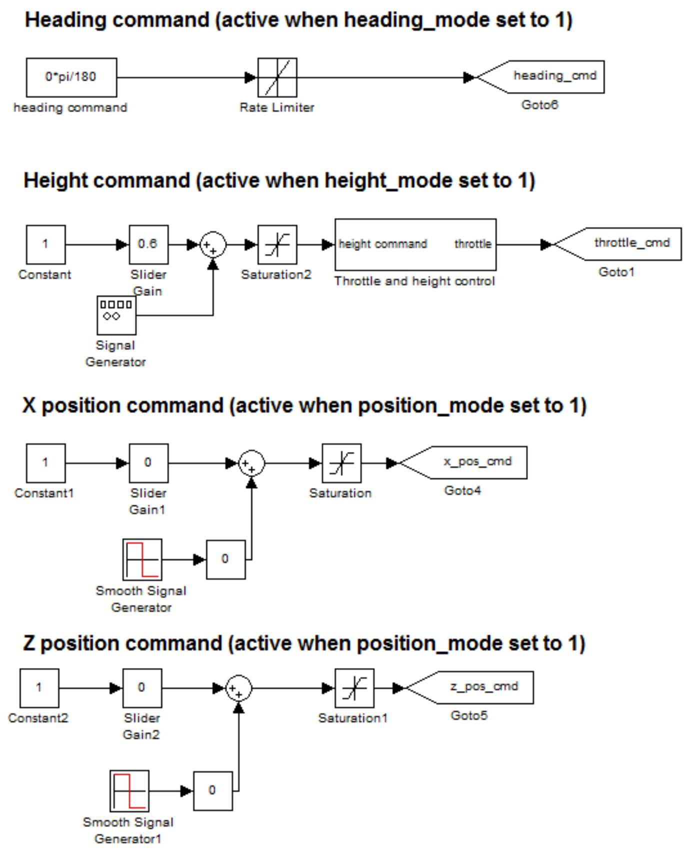

Figure 21. Original Position-Commands Subsystem

B. WAYPOINT MANAGEMENT

Earlier in this thesis, the author showed that the process of path generation results in establishing a sequence of discrete nodes on a map. The ILP process reads these nodes and output an optimal flight path that connects all nodes in a single loop. The nodes when ordered in sequence become waypoints for the Qball to follow.

Quanser's autonomous control scheme calls for the user to define the coordinates of the waypoint in all three Cartesian axes, and it will maneuver the Qball to that waypoint accordingly. This scheme takes in only a single-step input, however, whereas a continuous signal input is required to navigate the Qball through multiple waypoints. Besides following the coordinates of the waypoints, a truly autonomous control scheme needs to read and perform different mission tasks such as taking off, hovering or landing to be autonomous during the mission.

This research proposes the construct of a user-defined MATLAB function–the "Waypoint Manager" that executes continuously in the Simulink environment. To use the waypoint manager, the user first has to define several matrices to store coordinates of the waypoints and the required tasks at each waypoint. Real-time coding by Simulink then reads these matrices through the waypoint manager, which uses the switch function to determine the output command signals into the existing control scheme. Another critical input for the waypoint manager is the OptiTrack coordinates because the waypoint manager compares the real-time input against the waypoint coordinates in temporary memory. As long as the difference is within a defined tolerance level, the Qball will meet arrival condition and the waypoint manager will proceed to read the next elements in the coordinates and task matrices. The user can also input operating height if the user desires to move the Qball in a flat trajectory. Similarly, the user can use the waypoint manager to introduce loitering time at each waypoint through the Waypoint Manager. Figure 22 presents the waypoint manager algorithm.

Such a waypoint-management subsystem has the benefit of not imposing further constraints on the existing control scheme. Moreover, the user command occurs in state space, which is intuitive to the user, and the user can apply the output from the optimized search path directly.

```
    Input: commanded coordinates, OptiTrack coordinates, operating height, next task, simulation time
    Output: position commands

1.  tolerance = 0.1 m
2.  hover time = 5 sec
3.  index = 1
4.
5.  switch mission state
6.      case initialize
7.          position commands = 0
8.          if simulation time >= 3
9.              mission state = take off
10.         end if
11.     case takeoff
12.         position commands = [0, 0, operating height]
13.         ifabs(position commands - OptiTrack coordinates) < tolerance
14.             mission state = go to waypoint
15.         Endif
16.     case go to waypoint
17.         position commands = commanded coordinates (index)
18.         ifnext task = go on
19.             ifabs(position commands - OptiTrack coordinates) < tolerance
20.                 mission state = hover
21.                 time to move = simulation time + hover time
22.             Endif
23.         else
24.             mission state = land
25.         Endif
26.     case land
27.         position commands = [commanded coordinates(1), commanded coordinates(2), 0]
28.     case hover
29.         if simulation time >= time to move
30.             mission state = go to waypoint
31.             index = index + 1
32.         Endif
33. Endswitch
```

Figure 22. Waypoint Management Algorithm

C. THE COLLISION-AVOIDANCE SUBSYSTEM

While the previous control algorithm gives some control over the height management of the Qball by specifying the Z-coordinates or operating height, it does not yield an optimal trajectory that avoids intermediate obstacles between waypoints. Thus, there is a need to correct the trajectory between two waypoints to avoid known obstacles. Generating a near-optimal trajectory using the IDVD avails the possibility to achieve this correction. The previous chapter has explained the methodology of generating the near-optimal trajectory. The task remains to integrate the two control inputs.

To fulfil the above requirement, the author proposes a collision avoidance subsystem to input the control signal generated by the direct method at the known waypoint before the obstacle. A redlined box enclosed the subsystem in Figure 23. As the Qball arrives at this known waypoint, the prior-mentioned waypoint manager reads the next task as avoidance of imminent obstacle, instead of merely as “go on.” As such, the position commands will receive signal input from the newly introduced subsystem instead. After the Qball executes the obstacle avoidance maneuver and arrives at the next waypoint, readings from OptiTrack will cause the waypoint manager to switch back to reading control signals from the pre-defined waypoint coordinates and task matrices. The Qball then continues on its optimal search path.

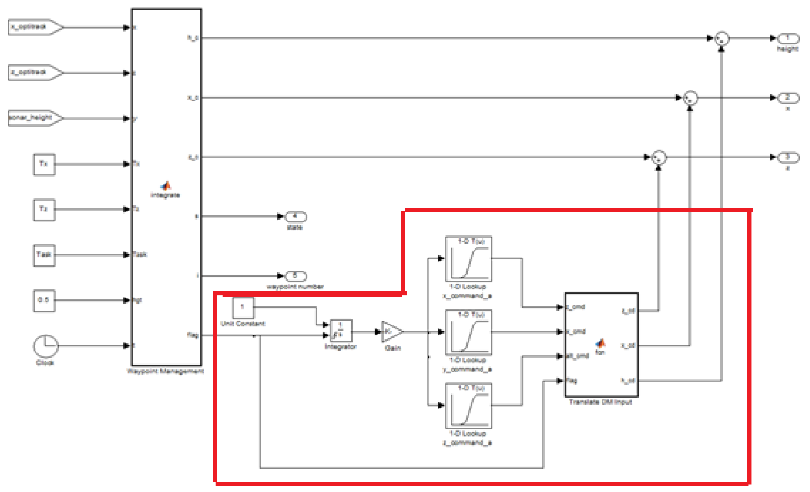

Figure 23. Collision Avoidance Subsystem as a Part of Waypoint Manager

The collision avoidance subsystem consists of three 1-D lookup tables that stores the interpolated coordinates along the near-optimal overhead trajectory. An adjacent user-defined MATLAB function rotates the Cartesian coordinates into the coordinate frame used by the control scheme. Close examination of the collision avoidance subsystem seems to indicate less-than-elegant modelling. This is largely due to the limitation of Simulink whereby it reads all input signals at the start of the simulation. To bypass this constraint, the subsystem uses the integrator block with a flat unit signal to prevent the output of the lookup table. When the Qball arrives at the waypoint immediately before the obstacle, the waypoint manager raises a flag. This sends a rising signal into the integrator block, causing the lookup table to re-output its stored data from the beginning at a rate that matches the size of the simulation time step. Hence, the state space where the Qball physically resides matches the initial input of the collision-avoidance signal and the Qball may smoothly carry out the overhead maneuver.

Upon the conclusion of the avoidance, the lookup table clips its output signal at the last line read; therefore, the user needs to set this final input to zero at the system-initialization phase. As a result, the subsystem adds only null to the waypoint signal as a command-position output. Another possible method explored in this research is to use time as a means of preventing any additional collision-avoidance and waypoint signals. Because the user knows the total time required to complete the overhead maneuver, owing to his prior input to the waypoint manager, the code can count the time elapsed during the maneuver and lower a flag at the end of the stated time to stop further output from the lookup table. Thus, only the waypoint signal creates position-command output. Both these methods work, with no significant difference in outcome.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. SIMULATION RESULTS

Given the limited flight experimentation area at ASEIL, the author modified and downscaled the mission scenario. The first experiment dealt with the Qball following a flat trajectory set by mere input of waypoint coordinates. The purpose is to verify the method of waypoint navigation using the existing controller. The second experiment observed the Qball executing a maneuver over an obstacle between two waypoints using output from the near-optimal trajectory generation. The intent here is to determine the ability of the controller to execute the trajectory generated using the IDVD method. The third experiment was devoted to the Qball following a flat trajectory first before executing an overhead maneuver in the middle of its mission path in a mixed position command signals scenario. The resolve in this experiment is to validate the architecture set up to perform collision avoidance amidst the search mission.

A. INITIAL SETUP PROCEDURES

Before conducting each experiment, the author used to the following checklist for initial setup:

- Charge two LiPo batteries to full capacity before establishing wired connection to Qball.
- Switch on Qball using the two switches along the power cable.
- Establish wireless connection between Qball and GCS on the wireless network and QUARC interface. The latter requires the input of the QBall's IP address as target uniform resource locator.
- Check Host Joystick model for up-to-date entry of calibration and trackable files.
- Check Qball Controller model for correct entry of host and target IP addresses.
- Load any necessary data from other codes or models, e.g., coordinates of waypoints.
- Perform incremental build on both Simulink models. Connect to target and start real-time coding on the Host Joystick model before Qball controller model.

B. CONDUCT OF EXPERIMENTS

1. Waypoint Following

In this experiment, the intent was to observe if the QBall was able to navigate itself successfully through all the defined waypoints in a sequential manner. In addition, the observer verified straight and level flight posture towards each waypoint.

a. Input parameters

The author established five waypoints, not including the start and endpoint, along planned the flight profile. Table 4 captures the X-Z coordinates of these waypoints, including the expected task. Given the limited flight experimentation area, a concern was that the waypoints might not be reasonably far apart; but the conditions proved satisfactory in providing respectable results. The author kept the trajectory flat by holding the operating height at 0.6 m. The Qball started its flight from the origin and passed each waypoint before moving onto the next waypoint. The Qball then landed at the final waypoint.

Table 4. Coordinates of Waypoints and Respective Task to Execute

Waypoint #	x (m)	z (m)	Task
0	0	0	Take off
1	-1	1	Go on
2	1	1	Go on
3	1	-1	Go on
4	-1	-1	Go on
5	-1	0	Go on
6	0	0	Land

b. Flight Results

The QBall executed a successful flight consistent with the waypoints and tasks allocated. From Figure 24, the Qball maintained a relatively flat trajectory as it flew along the path defined by the waypoints.

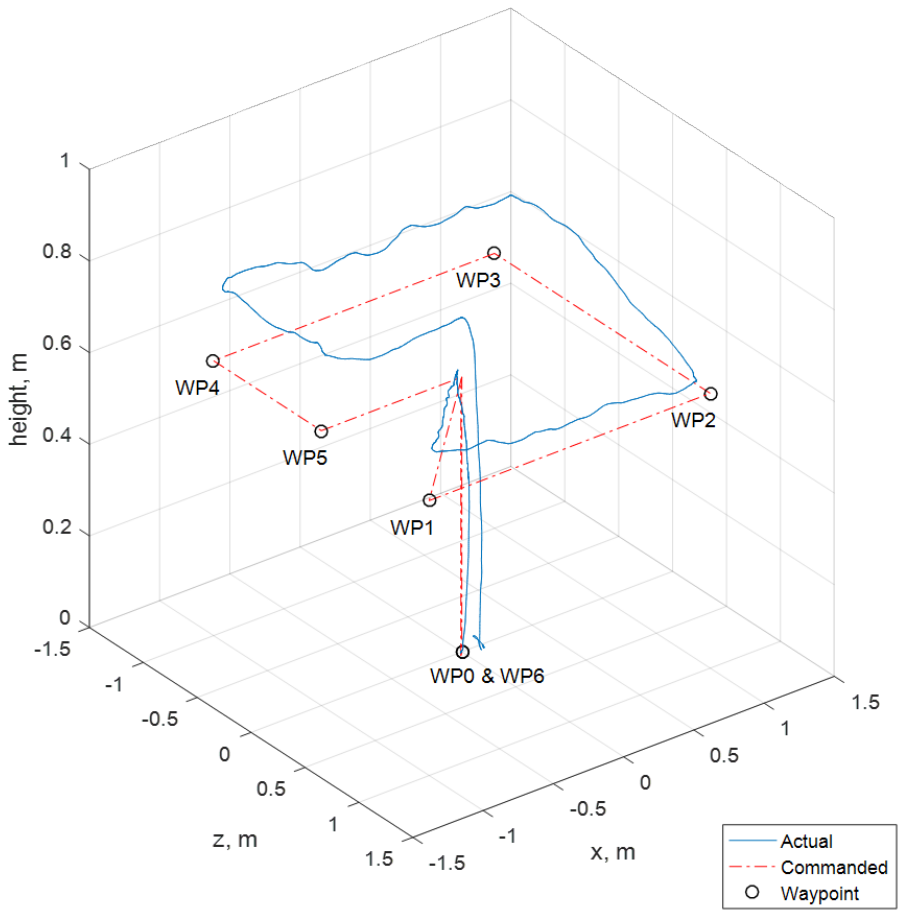

Figure 24. 3D View of the Executed Flight Profile

In Figure 25, one can see the flight trajectory of the Qball from the top view. The Qball maintained a relatively straight course towards each waypoint.

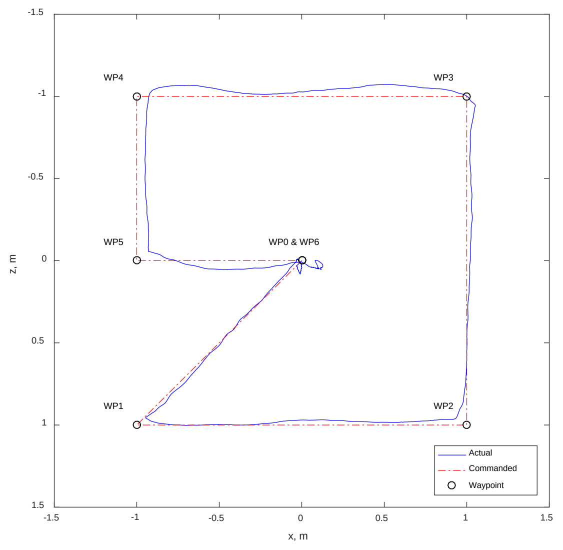

Figure 25. Top View of Qball’s Flight Trajectory

Figure 26 presents time histories of X, Y, Z coordinates along with commanded inputs. One can appreciate the fact that the Qball has achieved a close alignment to the commanded signal as the two lines nearly coincide in each axes. The third subplot, however, indicated some differences between the actual path and command signal of approximately 20 cm, due to small errors generated by the Quanser’s LQR controller.

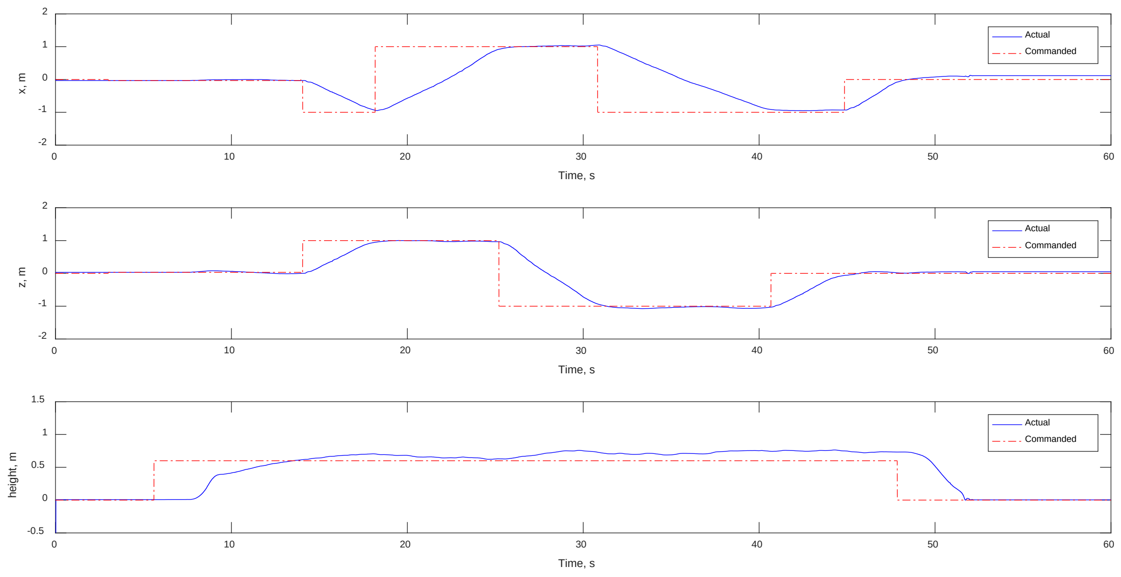

Figure 26. Time Histories of X, Y, Z Coordinates with Commanded Inputs

2. Obstacle Avoidance

This experiment assumed that the operator found an obstacle in the path of the quadrotor that exceeded the operating height of the mission. The obstacle had a height of 0.65m and an overhead maneuver was preplanned to avoid it. The author then generates a near-optimal trajectory using IDVD, and pipes the output to the "Position Command" Subsystem.

a. *Input parameters*

To generate the near-optimal trajectory using IDVD, it is necessary to spell out the boundary conditions of the system. These values as tabulated in Table 5 forms the input parameters to the trajectory optimization model.

Table 5. Input Parameter to Trajectory Optimization Model

Parameter	**Value**
Time given to execute maneuver, t_{des}, sec	20 sec
Initial waypoint coordinates, $x(t_0), y(t_0), z(t_0)$, m	1, 0, 0.6
Initial velocity, $\dot{x}(t_0), \dot{y}(t_0), \dot{z}(t_0)$, m/s	0
Initial acceleration, $\ddot{x}(t_0), \ddot{y}(t_0), \ddot{z}(t_0)$, m/s^2	0
Final waypoint coordinates $x(t_0), y(t_0), z(t_0)$, m	-1, 0, 0.6
Final velocity, $\dot{x}(t_0), \dot{y}(t_0), \dot{z}(t_0)$, m/s	0
Final acceleration, $\ddot{x}(t_0), \ddot{y}(t_0), \ddot{z}(t_0)$, m/s^2	0

b. *Model Output*

With the input parameters set into the trajectory optimization model, it output the near-optimal trajectory with interpolated waypoints between the initial and final waypoints. Figure 27 illustrates the side view of the planned maneuver. The area shaded in green represents the obstacle that is of height 0.65m. The area in yellow represents the safety buffer to account for any mid-flight disturbance or erroneous approximation of the actual height of the obstacle. Figure 28 presents a top view of the same maneuver.

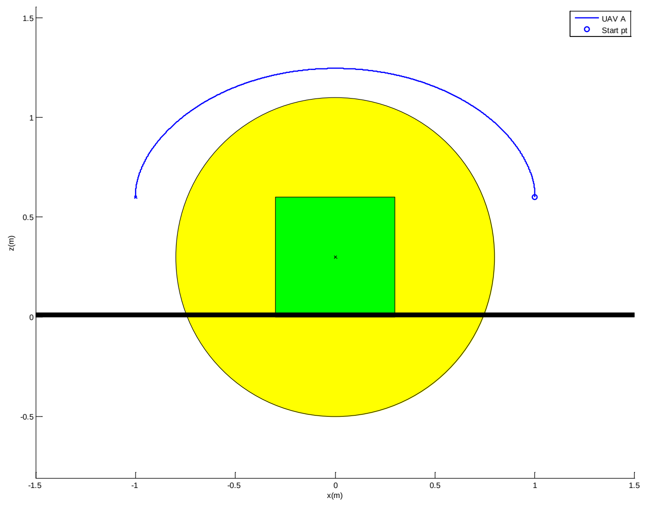

Figure 27. Near-Optimal Trajectory Generated Using IDVD to Avoid Vertical Obstacle (Side View)

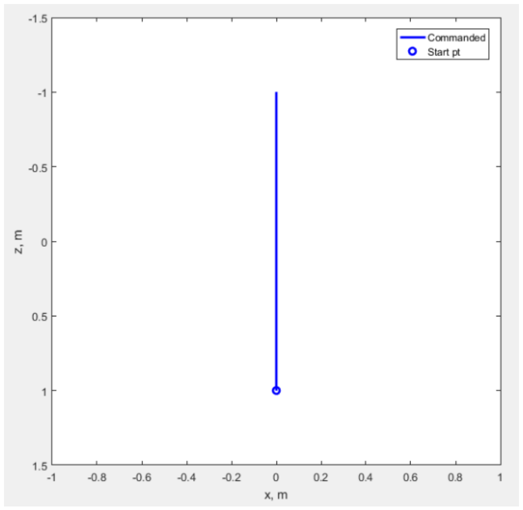

Figure 28. Near-Optimal Trajectory Generated Using IDVD to Avoid Obstacle (Top View)

c. *Flight Results*

The Qball again produced a successful flight using only the input of the interpolated waypoints, which serve as position commands for the control scheme. Figure 29 presents the actual trajectory undertaken by the Qball. The Qball adhered to the commanded path with minimal error. Figure 30 illustrates the flight path of the Qball from a top view. One can observe that the Qball has managed to fly in almost a straight line. Figure 31 presents the time histories of X, Y, Z coordinates along with commanded inputs. Again, the reader could see the Qball performing well in executing the commanded position control signal.

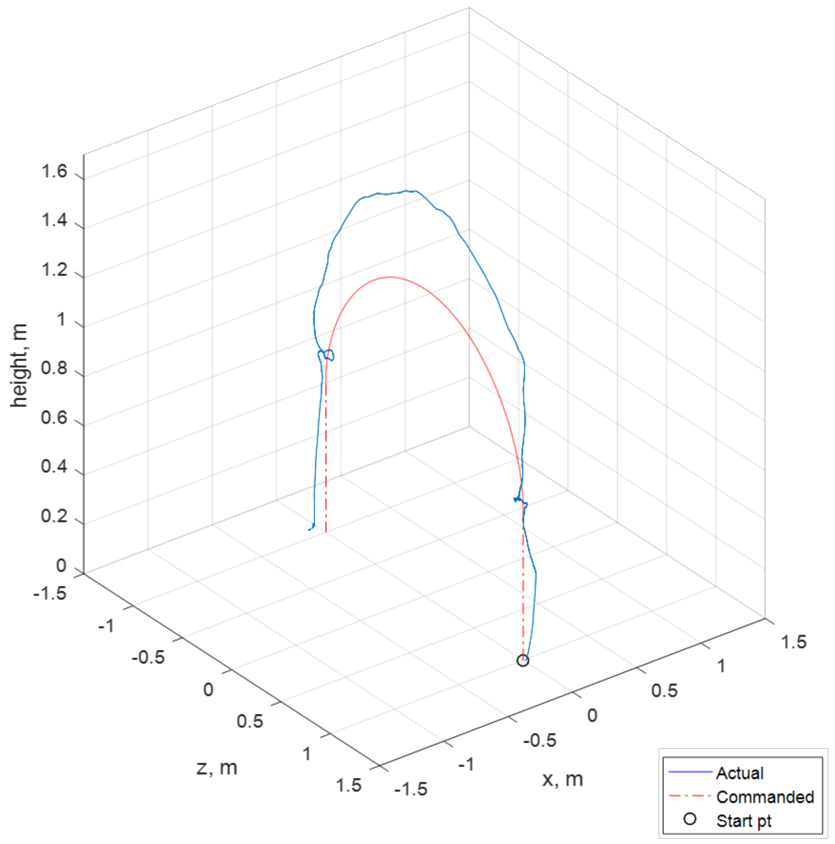

Figure 29. 3D View of the Executed Flight Profile

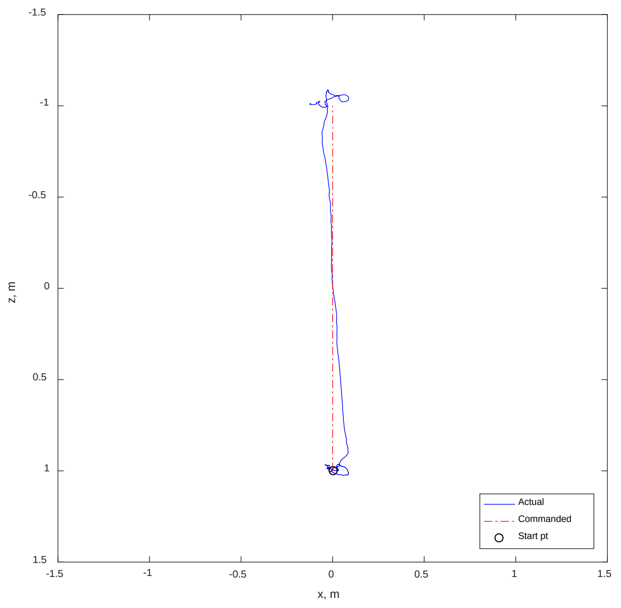

Figure 30. Top View of Qball's Flight Trajectory

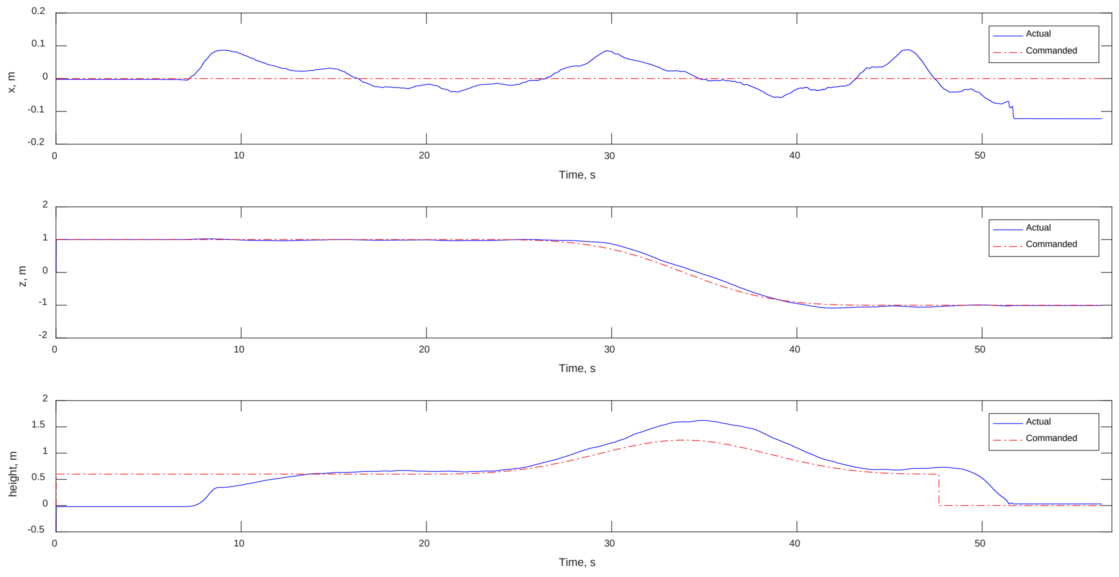

Figure 31. Time Histories of X, Y, Z Coordinates with Commanded Inputs

3. Mixed Guidance

For the mixed guidance experiment, the author uses the same set of waypoints established in the first experiment, with the exception of the task at waypoint 3, which the author changed to execute an overhead maneuver. The near-optimal trajectory generated in second experiment is still relevant with to this experiment with merely a translation of the position commands in the x-axis from origin to x = 1. Two position command signals the collision avoidance signal and waypoint signal are now both inputs into the Qball control scheme through the waypoint manager.

a. Flight Results

The experiment yielded the successful results seen in Figure 32. The Qball maintained a flat trajectory throughout its flight until arrival at the waypoint where it executed a successful overhead trajectory to avoid a vertical obstacle.

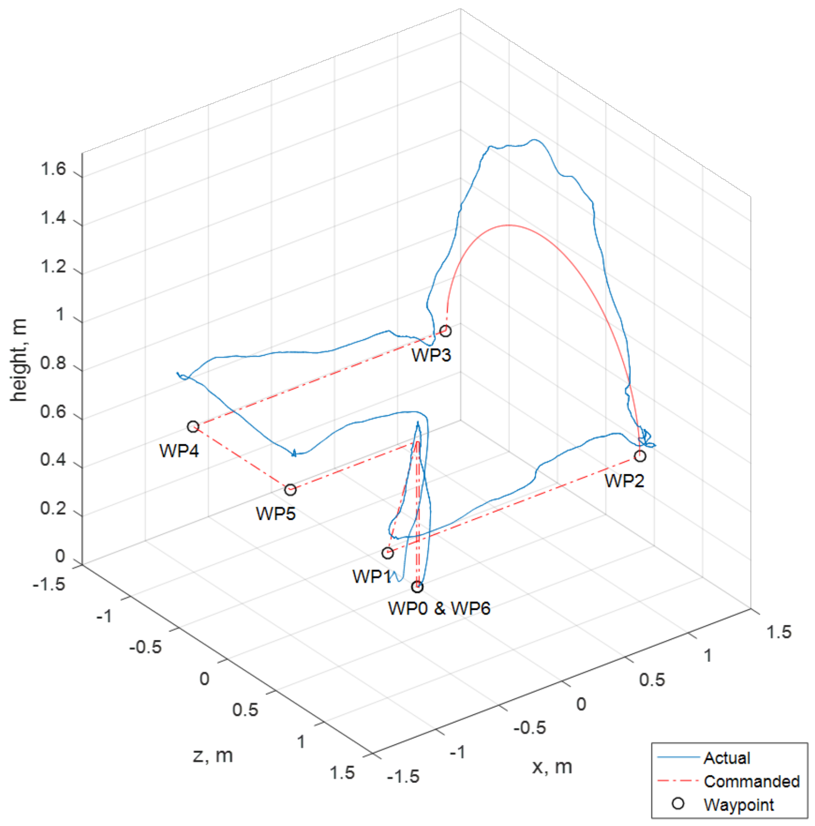

Figure 32. 3D View of Integrated Flight Profile

In Figure 33, one can observe the Qball's flight trajectory from the top view. The Qball executed an overhead maneuver between (1, 1) and (1, -1) in an otherwise flat trajectory. The outcome seems similar to that seen in the first experiment: this is a deliberate juxtaposition to emphasize that, even with the mixing of position-control commands, the QBall can follow and execute commands appropriately. The same effect is seen in Figure 34, which presents the time histories of the X, Y, and Z coordinates, along with commanded inputs. Again, the QBall performed well in the state space.

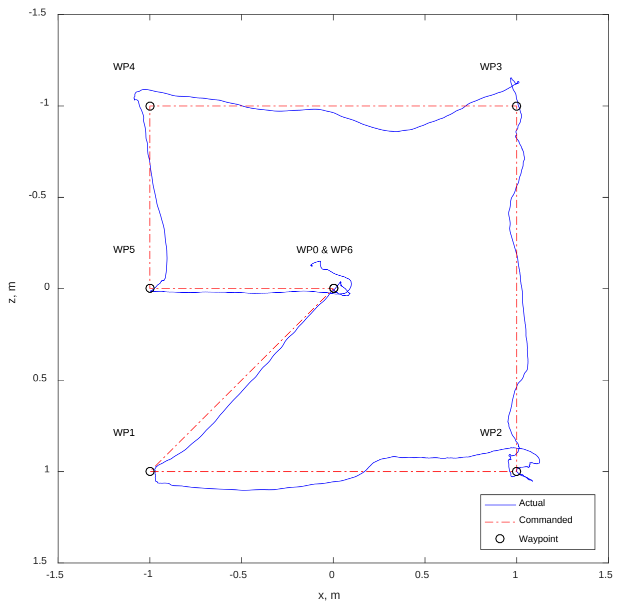

Figure 33. Top View of Qball's Trajectory in an Integrated Flight Profile

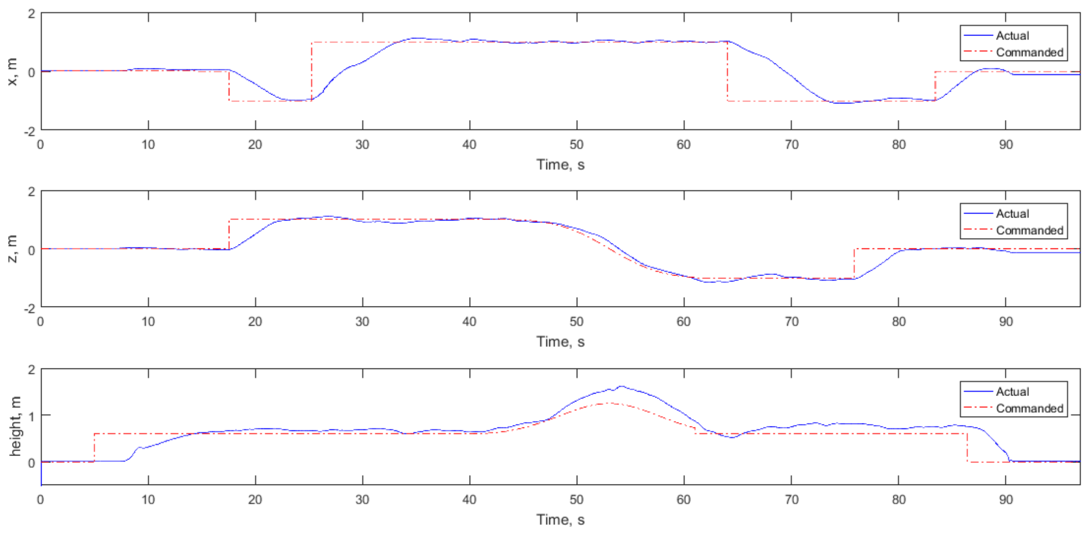

Figure 34. Time Histories of X, Y, Z Coordinates with Commanded Inputs

Figure 35 presents a side view of the Qball's trajectory. While the shape of the actual trajectory closely resembles the commanded trajectory, one can still observe some errors especially at the apogee. The Quanser's control scheme also takes in sonar readings from the sensor located at the base of the Qball to adjust its flight controls. The precision level of the sensor affects the readings, which in turn affects the vertical flight vector control. Nevertheless, the control scheme still manages to execute the maneuver as intended.

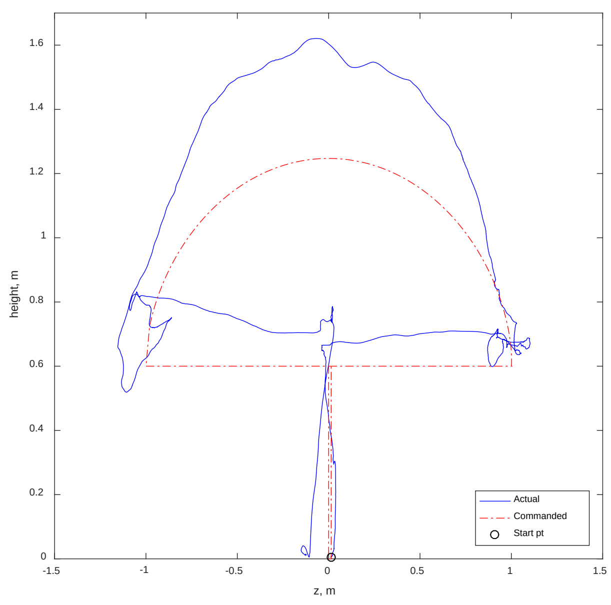

Figure 35. Side View of Qball's Trajectory

VIII. CONCLUSION AND RECOMMENDATIONS

A. CONCLUSION

This thesis introduces a feasible system architecture that uses visual sensors to achieve autonomous capabilities using quadrotors. By using the systems engineering approach, it had facilitated the autonomous system design through the examination of the stakeholders' needs and followed by the formulation of functional requirements. Such a method of system thinking provides a logical and comprehensive design solution.

Based on the system architecture proffered, the thesis proposes a means to further flight autonomy, providing verification through a series of simulations. Specifically, the system optimized the search path using ILP before carrying out waypoint navigation. With user's input of possible vertical obstacles, the system executed an overhead maneuver with the near-optimal trajectory generated using IDVD. The successful flight simulations demonstrated the feasibility of autonomous trajectory following and obstacle avoidance.

Problems with the software for a miniature IP camera prevented integration of the camera into the quadrotor, as originally planned. However, the author has developed onboard autopilot software in anticipation that live camera signals will be available in the near future as vendors correct the software problems.

The thesis demonstrated that a quadrotor-based autonomous system is a feasible and definitely useful option for the SAF advances in manned-unmanned systems integration. With an autonomous system, it could greatly enhance strike and reconnaissance capacity and capability, allowing the SAF to fortifying its strike competency by seeing first, seeing fast, and seeing more.

B. RECOMMENDATIONS

The author recommends the following future work in similar areas of interest:

- Embed visual sensors to achieve real-time obstacle detection for mid-flight collision avoidance.

- Enable the input of GPS coordinates into the control scheme so that future work may extend indoor experimentation to outdoor usage.
- Improve the feedback mechanism of the existing controller to minimize height errors between commanded and actual flight.
- Create a graphical user interface that enables the operator to perform the entire mission via a single interface, instead of the current piecemeal approach.

APPENDIX A. IMAGE PROCESSING FOR REAL-TIME TRACKING

The employment of cameras is inevitable in a target-acquisition system, especially so in an autonomous system. The computer perceives and processes the world through the camera, and therefore the discipline of computer vision comes into play as it attempts “to describe the world that we see in one or more images and to reconstruct its properties, such as shape, illumination, and color distributions” (Szeliski 2010, 5). While humans perform the task of visioning effortlessly, a computer takes a longer and uses more resources to interpret the same data—and yet its output tends to be error prone.

The achievement of computer vision depends on the type of camera used. While the common camera captures visible light to form an image, thermographic camera uses infrared radiation for imaging. Researchers and operators commonly apply both techniques of imaging in a single electro-optical/ infrared (EO/IR) camera module, as seen in Figure 36. These modules are high in both precision and resolution. While it is beyond the scope of this thesis to explore EO/IR imagery, the author used an Internet protocol (IP) camera to model and demonstrate the imaging functions that may enhance the overall capability of an autonomous target-acquisition quadrotor.

Figure 36. Star SAFIRE 380-HD. Source: FLIR Systems, Inc. (2016).

A. COLOR VS. GRAY-SCALE VIDEO

It is imperative to understand the basics of imagery prior to delving deeper into target detection as part of target acquisition. Starting from the typical video output of a surveillance camera, the video is in essence as a sequence of digital images woven together. Picture elements, or pixel in short make up a digital image; and a pixel is the most basic unit, which represents a color for a physical point in the digital space. The red-green-blue (RGB) color system can specify the color of this pixel. It expresses each of the three additive colors (red, green and blue), in 256 levels. The combination of three colors at any desired level forms a spectrum of color space, which contains over 16.7 million different possible colors.

Three matrices in the MATLAB environment can therefore represent a digital image, one matric for each of the three colors. All three matrices have the same number of elements as the number of pixels in that image, and each element contains the color level value of each pixel in the same physical space. Reading just a single matric would yield the same image in a monochromatic fashion.

While it is intuitive for a human to interpret images in color, doing likewise in the computer is computationally intensive given the number of elements to process. Three matrices in the RGB system represent a fully colored image, whereas the use of greyscale only needs only one matric for representation. Color information will be, however, lost in greyscale as only the intensity of white against black is stored. Nevertheless, there is a saving in computer memory storage space, and it facilitates faster processing subsequently as only one layer is required to be read. A binary image, which is also of a single layer but only has black and white color. As such, it is desirable to perform image processing in grey- or binary scale so that there is a shorter processing, which can be crucial in a time-sensitive environment.

B. BLOB ANALYSIS TO ATTAIN TARGET CENTROID

MATLAB’s computer-vision toolbox affords the user with many ready-to-use functions. One is blob analysis, which computes various “statistics for labeled regions in a binary image” (MATLAB 2016, 1–186). Of importance is the return of the centroid position of the blob. The author sets minimum and maximum values of 500 and 15000 respectively for detection of the desired blob. Such a setting is predefined and can be akin to the estimated target size with respect to the screen.

A necessary output of target acquisition is the location of the target in the physical space. The location of which will facilitate own forces to carry out follow-on strike in the attempt to persecute the target. This is achievable by measuring the target's centroid relative to the frame captured by the IP camera. Assuming that the user knows the position of the Qball, he can determine the location of the target by computing the offset from the center of the screen. The user would have previously calibrated this offset and hence the computation would provide the coordinates of the target.

C. THE TREK AI-BALL

The Trek Ai-Ball is an IP camera, shown in Figure 37 that allows the user to view and record video wirelessly. A commercial product from Trek 2000 International Ltd, the Ai-Ball extends image capturing and processing capabilities to the Qball's architecture. The Ai-Ball is compact and lightweight in design. Powered by a CR2 battery, the Ai-Ball weighs approximately 100g and is only 30mm in diameter and 35 mm long. Its wireless communication capability makes it well-suited to integrate with the Qball. Table 6 lists the critical specification.

Figure 37. Trek Ai-Ball

Table 6. Specification of Trek Ai-Ball. Source: Trek (2007).

Capability	**Specification**
Field of View	60 degrees
Resolution	VGA 640 x 480 pixel
Frame Rate	Up to 30 frames per second
Wireless Interface	IEEE 802.11b/g 2.4GHz ISM Band

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. OPTIMIZATION SCRIPT TO PROCESS SEARCH AREA AND DETERMINE SEARCH PATH

```
clc
clear all

% read pre-processed map (with pre-shaded areas) and convert to gray
scale
% to flatten the image
orig_map=imread('area_of_ops_overlay.jpg');    % filename of image
gray_map=rgb2gray(orig_map);

% To-search area represented by a binary matrix. Areas which are not
shaded
% out will be 0 and to-search area by 1.
size_gray_map=size(gray_map);

for x=1:size_gray_map(1)
    for y=1:size_gray_map(2)
        if gray_map(x,y)>200       % empirical treshold for non-grey
area
            bin_map(x,y)=1;
        else
            bin_map(x,y)=0;
        end
    end
end

% To create x- and y-vector that represent the x- and y-coordinates on
a
% cartesian plane starting at the origin based on size of the map read.

ind_k=0;
for ind_i=1:numel(bin_map)
    if mod(ind_i,size_gray_map(2))==0
        ind_k=ind_k+1;
    end
    y_vector(ind_i)=ind_k;
end

ind_n=0;
for ind_j=1:numel(bin_map)
    x_vector(ind_j)=ind_n;
    ind_n=ind_n+1;
    if ind_n==(size_gray_map(2))
        ind_n=0;
    end
end


%%
```

```
% To define a matrix, J that represents location of zeros (shaded area)
% within the binary matrix that represents the map
B=flipud(bin_map);
K=reshape(B’,1,[]);
J=find(K<1);

% To determine the x- and y-coordinates of the shaded area
gray_x=x_vector(J);
gray_y=y_vector(J);

% To remove the locations of the shaded area from x- and y-vector,
% resulting in the vector containing only the coordinates of the to-
search
% area
x_vector(J)=[];
y_vector(J)=[];

%%
% Given that the search camera has a certain Field of View (FOV),
assuming
% fixed in size, the to-search vectors can be reduced by eliminating
those
% coordinates that fall within the FOV when flying through an
aggregated
% point.
width=30;                           % FOV of camera

for ind_z=1:length(x_vector)
    if mod(x_vector(ind_z),width)>0
        x_vector(ind_z)=0;
    end
end
ind_u=find(x_vector<1);
x_vector(ind_u)=[];
y_vector(ind_u)=[];

for ind_z=1:length(y_vector)
    if mod(y_vector(ind_z),width)>0
        y_vector(ind_z)=0;
    end
end
ind_v=find(y_vector<1);
x_vector(ind_v)=[];
y_vector(ind_v)=[];

%%
% translation to account for different origin
origin=[-200 -200];       % user input
x_vector=x_vector+origin(1);
y_vector=y_vector+origin(2);
gray_x=gray_x+origin(1);
gray_y=gray_y+origin(2);

%% To generate a scatter plot that represents the to-search nodes
```

```
hold on
axis([origin(1) 640+origin(1) origin(2) 480+origin(2)]);
scatter(x_vector,y_vector,'.');
scatter(gray_x,gray_y,'.');

%% Finding the Optimized Flight Path

% To determine all possible combination of connecting the nodes so as
to
% measure the distance for each combination. Objective function would
be to
% minimize the total distance of the single path connecting all the
nodes.
num_nodes=numel(x_vector);
all_combi=nchoosek(1:num_nodes,2);
dist_x=x_vector(all_combi(:,1)) - x_vector(all_combi(:,2));
dist_y=y_vector(all_combi(:,1)) - y_vector(all_combi(:,2));
node_dist=sqrt(dist_x.^2+dist_y.^2);
len_node_dist=length(node_dist);

% Constraints for objective function:
% 1. All nodes are passed once
% 2. All nodes need to be connected

cons_1=spones(1:length(all_combi));
cons_2=num_nodes;

cons_1 =
[cons_1;spalloc(num_nodes,length(all_combi),num_nodes*(num_nodes-1))];
for node_ind = 1:num_nodes
    which_combi = (all_combi == node_ind);
    which_combi = sparse(sum(which_combi,2));
    cons_1(node_ind+1,:) = which_combi';
end
cons_2 = [cons_2; 2*ones(num_nodes,1)];

% To generate the binary decision variables
intcon = 1:len_node_dist;            % integer constraints
dec_zero = zeros(len_node_dist,1);   % lower bound
dec_one = ones(len_node_dist,1);     % upper bound

% Using intlinprog for as the optimization solver
opts = optimoptions('intlinprog','Display','off');
[path,costopt,exitflag,output] = intlinprog(node_dist,intcon,[],[],...
    cons_1,cons_2,dec_zero,dec_one,opts);

% Update Plot
hold on
segments = find(path);
line_handle = zeros(num_nodes,1);
line_handle = updatePlot(line_handle,path,all_combi,x_vector,y_vector);

% check for any sub-optimal solution, i.e., number of paths required to
% connect all the nodes
```

```
route_check = checkSub(path,all_combi);
num_sub_path = length(route_check);
if num_sub_path==1
    fprintf('Optimal path found.\n');
else
    fprintf('# of sub-optimal paths: %d\n',num_sub_path);
end

% Reiteration of solver until optimal path is found
Ineq_1 = spalloc(0,len_node_dist,0);
Ineq_2 = [];
while num_sub_path > 1
    Ineq_2 = [Ineq_2;zeros(num_sub_path,1)];
    Ineq_1 = [Ineq_1;spalloc(num_sub_path,len_node_dist,num_nodes)];
    for node_ind = 1:num_sub_path
        ind_row = size(Ineq_1,1)+1;
        ind_sub_path = route_check{node_ind};
        all_var = nchoosek(1:length(ind_sub_path),2);
        for ind_t = 1:length(all_var)
            whichVar =
(sum(all_combi==ind_sub_path(all_var(ind_t,1)),2))...
                & (sum(all_combi==ind_sub_path(all_var(ind_t,2)),2));
            Ineq_1(ind_row,whichVar) = 1;
        end
        Ineq_2(ind_row) = length(ind_sub_path)-1;
    end

    % Recall Solver
    [path,costopt,exitflag,output] =
intlinprog(node_dist,intcon,Ineq_1,...
        Ineq_2,cons_1,cons_2,dec_zero,dec_one,opts);

    % Update Plot
    line_handle =
updatePlot(line_handle,path,all_combi,x_vector,y_vector);

    % Check for sub-optimal path
    route_check = checkSub(path,all_combi);
    num_sub_path = length(route_check); % number of subtours
    if num_sub_path==1
        fprintf('Optimal path found.\n');
    else
        fprintf('# of sub-optimal paths: %d\n',num_sub_path);
    end
end

%% to determine coordinates of waypoints in sequence

% route row index shows sequence
% route(:,1) shows waypoint number in graph
% route(:,2) shows x-coord for waypoint
% route(:,3) shows y-coord for waypoint

route_seq(:,1)=route_check{1}';
```

```
for ind_a=1:num_nodes
    route_seq(ind_a,2)=x_vector(route_seq(ind_a,1));
    route_seq(ind_a,3)=y_vector(route_seq(ind_a,1));
end
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Blanchard, Benjamin S. and Wolter J. Fabrycky. 2011. *Systems Engineering and Analysis*. 5th ed., Boston: Pearson.

Boyle, Rebecca. 2012. “One Third of the Military’s Aircraft are Now Drones.” *Popular Science*, January 10. http://www.popsci.com/technology/article/2012-01/drones-now-account-nearly-one-third-militarys-aircraft.

Chua, Boon Heng. 2013. “Integration of Multiple Unmanned Systems in an Urban Search and Rescue Environment.” Master’s thesis, Naval Postgraduate School.

Cook, K. L. B. 2007. “The Silent Force Multiplier: The History and Role of UAVs in Warfare.” In 2007 *IEEE Aerospace Conference*. doi 10.1109/AERO.2007.352737

Cowling, Ian D., James F.Whidborne, and Alastair K. Cooke. 2006. “Optimal Trajectory Planning and LQR Control for A Quadrotor UAV.” In *UKACC International Conference on Control*.

Cowling, Ian D., Oleg A. Yakimenko, James F. Whidborne, and Alastair, K. Cooke. 2010. “Direct Method Based Control System for an Autonomous Quadrotor.” *Journal of Intelligent and Robotic Systems* 60 (2): 285–316. doi:10.1007/s10846-010-9416-9.

Defence Media Centre. 2008. “Factsheet: STORM - STrike ObserveRs Mission.” https://www.mindef.gov.sg/imindef/press_room/official_releases/nr/2008/may/18may08_nr/18may08_fs2.html., last modified May 18. Singapore: Ministry of Defence

———. 2011. “Fact Sheet: High Mobility Artillery Rocket System (05 Sep 11).” https://www.mindef.gov.sg/imindef/press_room/official_releases/nr/2011/sep/05sep11_nr/05sep11_fs.html., last modified Sep 5. Singapore: Ministry of Defence

FLIR Systems, Inc. 2016. “Star SAFIRE 380-HD.” Accessed August 12. http://www.flir.com/surveillance/display/?id=64812

Google Maps. 2016. “The Great Lakes.” Accessed August 20. https://www.google.com/maps/place/The+Great+Lakes/@45.0426169,-88.6972133,6z/data=!3m1!4b1!4m5!3m4!1s0x4d4563ce559b5571:0x57688a459e8cac9b!8m2!3d45.0522366!4d-82.4846115

Guizzo, Erico. 2011. “Aeryon Scout Quadrotor Spies on Bad Guys from Above.” *IEEE Spectrum*, May 6. http://spectrum.ieee.org/automaton/robotics/military-robots/aeryon-scout-quadrotor-spies-on-bad-guys-from-above.

Haskins, Cecilia, Kevin Forsberg, and Michael Krueger. 2006. *Systems Engineering Handbook*. 3 ed.. Seattle, WA: International Council on Systems Engineering.

Honour, Eric C. 2004. “Understanding the Value of Systems Engineering.” *INCOSE International Symposium* 14 (1): 1207–1222. doi:10.1002/j.2334-5837.2004.tb00567.x.

Joint Chiefs of Staff. 2007. *Joint Targeting* (JP 3–60). Washington, DC: Department of Defense.

Koo, T. John, and Shankar Sastry. 1999. "Differential flatness based full authority helicopter control design." In *Decision and Control*, 1999. *Proceedings of the 38th IEEE Conference on*, vol. 2, pp. 1982-1987. IEEE.

LaValle, Steven M. 2006. *Planning Algorithms*. New York, USA: Cambridge University Press.

Lin, Patrick, George Bekey, and Keith Abney. 2008. *Autonomous Military Robotics: Risk, Ethics, and Design*. California Polytechnic State Univ San Luis Obispo.

MATLAB. 2016. *Computer Vision System Toolbox Reference*. Natick, MA: The MathWorks, Inc.

Office of the Secretary of Defense. 2005. *Unmanned Aircraft Systems Roadmap 2005 – 2030*. Washington, DC: Department of Defense.

OptiTrack. 2014. *Flex 3 Data Sheet*. NaturalPoint, Inc. https://www.optitrack.com/static/documents/Flex%203%20Data%20Sheet.pdf

Quanser. 2010. *Quanser Qball-X4, User Manual*. Canada: Quanser, Inc.

———. 2016. *QUARC Real-Time Control & Rapid Prototyping Software for MATLAB/Simulink*. Quanser, Inc. http://www.quanser.com/Products/Docs/3807/QUARC_Product_Info_Sheet_v3.pdf

Saad, Imelda. 2011. “SAF Uses Technology to Counter Lower Birth Rate & Ageing Population.” *Channel NewsAsia*, November 12. http://www.channelnewsasia.com/news/singapore/saf-uses-technology-to-co/514780.html.

Shachtman, Noah and Spencer Ackerman. 2012. “Almost 1 in 3 U.S. Warplanes is a Robot.” *Wired*. https://www.wired.com/2012/01/drone-report/.

Singapore Department of Statistics. 2015. Population Trends 2015. https://www.singstat.gov.sg/docs/default-source/default-document-library/publications/publications_and_papers/population_and_population_structure/population2015.pdf.

Skolnick, Fred R. and Phillip G. Wilkins. 2000. "Laying the Foundation for Successful Systems Engineering." *Johns Hopkins APL Technical Digest* 21 (2): 208–216.

Szeliski, Richard. 2010. *Computer Vision: Algorithms and Applications*. New York: Springer.

Teo, Jing Ting. 2016. "SAF 2030 Sneak Peek." *Cyber Pioneer*, February 1. https://www.mindef.gov.sg/imindef/resourcelibrary/cyberpioneer/topics/articles/features/2016/feb16_fs1.html#.V4_aqbiAOkp.

Trek. 2007. *Operating the Ai-Ball*. Singapore: Trek 2000 International Ltd.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California